

19



OFICINA ESPAÑOLA DE
PATENTES Y MARCAS

ESPAÑA



11 Número de publicación: **2 937 793**

51 Int. Cl.:

H04N 19/91 (2014.01)
H04N 19/13 (2014.01)
H04N 19/70 (2014.01)
H04N 19/46 (2014.01)
H04N 19/463 (2014.01)
H04N 19/174 (2014.01)
H04N 19/436 (2014.01)

12

TRADUCCIÓN DE PATENTE EUROPEA

T3

96 Fecha de presentación y número de la solicitud europea: **15.04.2013** **E 20190474 (5)**

97 Fecha y número de publicación de la concesión europea: **09.11.2022** **EP 3793200**

54 Título: **Codificación de imagen con bajo retardo**

30 Prioridad:

13.04.2012 US 201261624098 P
29.06.2012 US 201261666185 P

45 Fecha de publicación y mención en BOPI de la
traducción de la patente:
31.03.2023

73 Titular/es:

GE VIDEO COMPRESSION, LLC (100.0%)
1 Research Circle
Niskayuna, NY 12309, US

72 Inventor/es:

SCHIERL, THOMAS;
GEORGE, VALERI;
HENKEL, ANASTASIA;
MARPE, DETLEV;
GRÜNEBERG, KARSTEN y
SKUPIN, ROBERT

74 Agente/Representante:

ARIZTI ACHA, Monica

ES 2 937 793 T3

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

DESCRIPCIÓN

Codificación de imagen con bajo retardo

- 5 La presente invención se refiere a la codificación de imágenes con bajo retardo.

En los sectores del diseño de HEVC actuales, los mosaicos de sectores de entropía (anteriormente sectores de peso ligero) y WPP (procesamiento en paralelo de frente de onda) están contenidos como herramientas para paralelización.

- 10 Para la paralelización de codificadores y decodificadores de video, la división a nivel de instantánea tiene varias ventajas en comparación con otros enfoques. En los códecs de video previos, como H.264/AVC [1], las divisiones de instantánea eran solamente posibles con sectores regulares con un alto coste en términos de eficacia de codificación. Para la decodificación de H.264/AVC en paralelo escalable es necesario combinar el paralelismo a nivel de macrobloque para la reconstrucción de instantánea y paralelismo a nivel de fotograma para decodificación de entropía.
- 15 Este enfoque, sin embargo, proporciona reducción limitada en latencias de instantánea y alto uso de memoria. Con el fin de superar estas limitaciones, nuevas estrategias de división de instantánea han sido incluidas en el códec de HEVC. La versión actual del software de referencia (HM-6) contiene 4 enfoques diferentes: sectores regulares o normales, sectores de entropía, sub-flujos de procesamiento en paralelo de frente de onda (WPP) y mosaicos. Típicamente, estas divisiones de instantánea comprenden un conjunto de unidades de codificación más grandes (LCU), o en una terminología sinónima, unidades de codificación en árbol (CTU), como se define en HEVC o incluso un subconjunto de aquellas.
- 20

- La figura 1 muestra una instantánea 898 colocada a modo de ejemplo en el sector regular 900 por fila 902 de LCU o macrobloques en una instantánea. Los sectores regulares o normales (como se define en H.264 [1]) tienen la penalización de codificación más grande ya que rompen las dependencias de codificación y predicción de entropía.
- 25

Los sectores de entropía, como los sectores, rompen las dependencias de decodificación de entropía pero permiten la predicción (y filtración) para cruzar límites de sector.

- 30 En WPP las divisiones de instantánea se intercalan por filas y se permite que tanto la decodificación como predicción de entropía usen datos de bloques en otras divisiones. De esta manera, las pérdidas de codificación se minimizan, mientras que al mismo tiempo el paralelismo de frente de onda puede aprovecharse. La intercalación, sin embargo, viola la causalidad de flujo de bits ya que una división anterior necesita una siguiente división para decodificar.

- 35 La figura 2 muestra a modo de ejemplo una instantánea 898 dividida en hasta dos filas 904, 904b de mosaicos de división horizontal 906. Los mosaicos definen límites horizontales 908 y verticales 910 que dividen una instantánea 898 en columnas 912a,b,c y filas 904a,b de mosaico. Similar a los sectores regulares 900, los mosaicos 906 rompen las dependencias de decodificación y predicción de entropía, pero no requieren un encabezamiento para cada mosaico.
- 40

Para cada una de estas técnicas, el número de divisiones puede elegirse libremente por el codificador.

- En general tener más divisiones conduce a pérdidas de compresión más altas. Sin embargo, en WPP la propagación de pérdida no es tan alta y por consiguiente el número de divisiones de instantánea incluso puede fijarse a una por fila. Esto conduce también a varias ventajas. En primer lugar, se garantiza la causalidad para el flujo de bits de WPP. En segundo lugar, las implementaciones de decodificador pueden suponer que está disponible una cierta cantidad de paralelismo, lo que también se incrementa con la resolución. Y, finalmente, ninguna de las dependencias de selección y predicción de contexto tienen que romperse en la decodificación en el orden de frente de onda, dando como resultado bajas pérdidas de codificación.
- 45

- 50 Sin embargo, hasta ahora toda la codificación en paralelo en los conceptos de transformada falla al proporcionar un logro de eficacia de alta compresión en combinación con mantener el bajo retardo. Esto también se mantiene para el concepto de WPP. Los sectores son las unidades más pequeñas para transporte, en la tubería de codificación y varios subflujos de WPP todavía tienen que transportarse en serie.
- 55

- En el documento MAURICIO ÁLVAREZ-MESA *et al.* Parallel video decoding in the emerging HEVC standard, 201.2 IEEE, INTERNATIONAL CONFERENCE ON ACOUSTICS, SPEECH AND SIGNAL PROCESSING (CONFERENCIA INTERNACIONAL SOBRE PROCESAMIENTO ACÚSTICO, DE VOZ Y DE SEÑAL) (ICASSP 2012): KIOTO, JAPÓN, 25 – 30, páginas 1545-1548, y en el documento Mirsa K. *et al.* 'Entropy slices for parallel entropy coding', 94. MPEG MEETING; 11-10-2010 - 15-10-2010; CANTÓN; (GRUPO DE EXPERTOS EN IMAGEN EN MOVIMIENTO O ISO/ IEC JTC1/SC29/WG11), n.º M18297, 2 de octubre de 2010 (02-10-2010), se presenta una estrategia de paralelización para la norma de codificación de video HEVC. La estrategia propuesta se basa en sectores de entropía que permiten aprovechar el paralelismo en la etapa de decodificación de entropía al tiempo que se mantiene una alta eficacia de codificación. El enfoque requiere codificar videos con un sector de entropía por fila de
- 60

LCU con el fin de decodificar múltiples filas de LCU de manera paralela al frente de onda.

Por consiguiente, es el objeto de la presente invención proporcionar un concepto de codificación de instantánea que permita la decodificación en paralelo según, por ejemplo, el procesamiento en paralelo de frente de onda con eficacia incrementada, tal como reduciendo aún más el retardo de extremo a extremo o mejorando la eficacia de codificación reduciendo la tara de codificación a gastarse.

Este objeto se consigue por la materia objeto de las reivindicaciones independientes.

Es un hallazgo básico de la presente invención que los conceptos de procesamiento en paralelo tales como procesamiento en paralelo de frente de onda, pueden realizarse con retardo de extremo a extremo reducido si el concepto de sector habitual según el cual los sectores se codifican/decodifican completamente independientemente de áreas de la instantánea fuera del sector respectivo, o al menos independientemente de áreas fuera del sector respectivo en cuanto a lo que se refiere a la codificación de entropía en favor de sectores de diferentes modos, en concreto los denominados sectores dependientes que permiten interdependencias a través de los límites de sector y otros que no, llamados sectores normales, por ejemplo.

Es un hallazgo básico adicional de la presente invención que puede combinarse con el primero o usarse de manera individual, que el concepto de procesamiento de WPP puede realizarse más eficazmente si las porciones de sintaxis de inicio de sector se usan para ubicar puntos de entrada de WPP.

Se describe a continuación realizaciones preferidas de la presente invención con respecto a las figuras, en las que realizaciones ventajosas son el objeto de las reivindicaciones dependientes. Entre las figuras,

- 25 La figura 1 muestra una instantánea dividida a modo de ejemplo en un sector regular por fila de LCU o macrobloques en una instantánea;
- La figura 2 muestra una instantánea dividida a modo de ejemplo en hasta dos filas de mosaicos divididos horizontalmente;
- 30 La figura 3 muestra a modo de ejemplo una asignación de divisiones codificadas en paralelo a un sector o segmento de transporte de red;
- La figura 4 muestra un diagrama esquemático que ilustra una fragmentación genérica de un fotograma con un enfoque de codificación de mosaico para un retardo de extremo a extremo mínimo;
- 35 La figura 5 muestra un diagrama esquemático que ilustra una fragmentación a modo de ejemplo de un fotograma con enfoque de codificación de WPP para un retardo de extremo a extremo mínimo;
- 40 La figura 6 muestra un diagrama de bloques esquemático que ilustra un escenario de una conversación utilizando servicios de video;
- La figura 7 ilustra esquemáticamente una posible planificación de tiempo de codificación, transmisión y decodificación para mosaicos con subconjuntos genéricos con retardo de extremo a extremo mínimo;
- 45 La figura 8 muestra esquemáticamente una planificación de temporización que obtiene típicamente un retardo de extremo a extremo;
- 50 La figura 9 ilustra una instantánea que tiene a modo de ejemplo bloques de árbol de codificación 11x9, que está dividida en dos sectores;
- La figura 10 ilustra una instantánea que tiene a modo de ejemplo bloques de árbol de codificación 13x8, que está dividida en tres mosaicos;
- 55 La figura 11 muestra un ejemplo para una sintaxis de conjunto de parámetros de secuencia;
- La figura 12 muestra un ejemplo para una sintaxis de conjunto de parámetro de instantánea;
- 60 La figura 13 muestra un ejemplo para una sintaxis de encabezamiento de sector;
- La figura 14 ejemplifica una división de una instantánea para procesamiento de WPP en un sector regular y para un procesamiento de bajo retardo, en sectores dependientes;

	La figura 15	muestra un ejemplo para una porción dentro de una sintaxis de conjunto de parámetro de instantánea;
5	La figura 16	muestra una posible sintaxis de encabezamiento de sector;
	La figura 17	ilustra esquemáticamente interdependencias de codificación para sectores normales (y sectores dependientes);
10	La figura 18	muestra un diagrama esquemático que compara una codificación para el transporte de bajo retardo de mosaicos (procesamiento en paralelo de frente de onda utilizando sectores dependientes);
15	La figura 19	ilustra una planificación de temporización que ilustra una codificación de WPP a modo de ejemplo con transmisión de bajo retardo de tubería cuando se usa el procesamiento en paralelo de frente de onda utilizando sectores dependientes como se muestra en el lado derecho de la figura 18;
20	La figura 20	muestra un diagrama esquemático que ilustra una mejora de robustez al usar sectores regulares como anclas;
	La figura 21	muestra otra realización para una sintaxis de encabezamiento de sector;
25	La figura 22	muestra otra realización para una sintaxis de conjunto de parámetro de instantánea;
	La figura 23	muestra un diagrama esquemático que ilustra un proceso de inicialización de probabilidad de símbolo para un sector dependiente en el caso de comenzar en el límite de instantánea izquierda;
30	La figura 24	muestra un diagrama esquemático de un decodificador;
	La figura 25	muestra esquemáticamente un diagrama de bloques de un decodificador junto con ilustrar esquemáticamente la división de una instantánea en bloques de codificación y sectores;
35	La figura 26	muestra esquemáticamente un diagrama de bloques de un codificador;
	La figura 27	muestra esquemáticamente una instantánea dividida en sectores normales y dependientes, llamados en este punto segmentos de sector;
40	Las figuras 28a y 28b	muestran esquemáticamente una instantánea dividida en sectores normales y dependientes llamados en este punto segmentos de sector, por una parte y mosaicos por otra parte;
45	La figura 29	muestra un diagrama de flujo que ilustra un proceso de inicialización de contexto utilizando sectores dependientes;
	La figura 30	muestra un diagrama de flujo que ilustra un proceso de almacenamiento de contexto para usar sectores dependientes; y
50	La figura 31	muestra esquemáticamente diferentes posibilidades de señalización de puntos de entrada de WPP.

A continuación, la descripción comienza con una descripción de los conceptos de hoy en día para permitir el procesamiento de instantánea en paralelo y codificación de bajo retardo, respectivamente. Se resumen los problemas que tienen lugar cuando se desea tener ambas capacidades. En particular, como se apreciará del siguiente análisis, el concepto de subflujo de WPP como se enseña hasta ahora entra de una manera en conflicto con el deseo de tener

55 el concepto de subflujo de WPP como se enseña hasta ahora entra de una manera en conflicto con el deseo de tener un bajo retardo debido a la necesidad de transportar subflujos de WPP agrupando los mismos en un sector. Las siguientes realizaciones presentan conceptos de procesamiento en paralelo tales como el concepto de WPP, aplicable a aplicaciones que necesitan aún menos retardo al ampliar el concepto de sector, es decir al introducir otro tipo de sector, más tarde llamados sectores dependientes.

60 La minimización del retardo de video de extremo a extremo de capturar para visualizar es uno de los objetivos principales en aplicaciones tales como conferencia de video y similares.

La cadena de procesamiento de señal para la transmisión de video digital consiste en cámara, dispositivo de captura,

codificador, encapsulación, transmisión, demultiplexor, decodificador, presentador y visualización. Cada una de estas etapas contribuye al retardo de extremo a extremo almacenando en memoria intermedia datos de imagen antes de su transmisión en serie a la etapa siguiente.

- 5 Algunas aplicaciones requieren la minimización de tal retardo, por ejemplo, manipulando remotamente objetos en áreas peligrosas, sin vista directa al objeto manipulado o cirugía invasiva mínima. Incluso un retardo corto puede dar como resultado dificultades graves de manipulación apropiada o incluso conducir a errores catastróficos.

- 10 En muchos casos, todo un fotograma de video se almacena en memoria intermedia en una etapa de procesamiento, por ejemplo, para permitir el procesamiento intra fotograma. Algunas etapas reúnen datos con el fin de formar paquetes que se reenvían a la siguiente etapa. En general, hay un límite inferior para el retardo que resulta de los requisitos del procesamiento local. Esto se analiza para cada etapa individual en más detalle a continuación.

- 15 El procesamiento dentro de la cámara no requiere necesariamente procesamiento de señal intra-fotograma, de tal manera que se proporciona retardo mínimo por el tiempo de integración del sensor, que está limitado por la velocidad de fotograma y algunas elecciones de diseño por el fabricante de software. La salida de la cámara está relacionada típicamente con el orden de exploración que normalmente inicia el procesamiento en la esquina superior izquierda, se mueve a la esquina superior derecha y continúa línea a línea a la esquina inferior derecha. En consecuencia, toma alrededor de un fotograma de duración hasta que todos los datos se transfieren del sensor a la salida de la cámara.

- 20 El dispositivo de captura podría enviar los datos de la cámara inmediatamente después de la recepción; sin embargo, típicamente almacenará en memoria intermedia algunos datos y generará ráfagas con el fin de optimizar el acceso de datos a la memoria o almacenamiento. Además, la conexión entre la cámara/capturador y la memoria del ordenador es típicamente limitante de la tasa de bits para reenviar los datos de instantánea capturados a la memoria para procesamiento adicional (codificación). Típicamente, las cámaras se conectan mediante USB 2.0 o pronto con USB 25 3.0, que siempre incluirá un transporte parcial de los datos de instantánea al codificador. Esto limita la capacidad de paralelización en el lado del codificador en escenarios de bajo retardo extremos, es decir, el codificador intentará iniciar la codificación tan pronto como sea posible, cuando los datos se hacen disponibles de la cámara, por ejemplo, en un orden de exploración de tramas de arriba a abajo de la imagen.

- 30 En el codificador, hay algunos grados de libertad que permiten una solución intermedia de eficacia de codificación, en términos de velocidad de datos necesarios para una cierta fidelidad de video, para reducción del retardo de procesamiento.

- 35 El codificador usa datos que ya se han enviado para predecir la imagen a codificarse posteriormente. En general, la diferencia entre la imagen real y la predicción puede codificarse con menos bits que los que serían necesarios sin la predicción. Estos valores de predicción necesitan estar disponible en el decodificador, por lo tanto, la predicción está basada en porciones previamente decodificadas de la misma imagen (predicción de intra-fotograma) o en otras imágenes (predicción inter-fotograma) que se han procesado anteriormente. Los estándares de codificación de video 40 de antes de HEVC usan solamente la parte de la imagen por encima de o en la misma línea, pero dejan, que se han codificado previamente, para predicción de intra-fotograma, la predicción de vector de movimiento y codificación de entropía (CABAC).

- Además de la optimización de la estructura de predicción, puede considerarse la influencia del procesamiento en paralelo. El procesamiento en paralelo requiere la identificación de áreas de instantánea que pueden procesarse 45 independientemente. Por razones prácticas, se escogen las regiones contiguas tales como rectángulos horizontales o verticales, que se llaman frecuentemente "mosaicos". En el caso de restricciones de bajo retardo, estas regiones deben permitir la codificación paralelizada de los datos que entran del capturador a la memoria, tan pronto como sea posible. Suponiendo una transferencia de memoria de exploración de trama, las divisiones verticales de los datos sin procesar 50 tienen sentido, con el fin de iniciar la codificación inmediatamente. Dentro de tales mosaicos, lo que divide la instantánea en divisiones verticales (consúltese con la figura a continuación), la intra-predicción, predicción de vector en movimiento y codificación de entropía (CABAC) pueden conducir a eficacia de codificación razonable. Con el fin de minimizar el retardo, solamente parte de la instantánea, partiendo de la parte superior, se transferiría a la memoria de fotograma del codificador y el procesamiento en paralelo debería iniciarse en los mosaicos verticales.

- 55 Otra manera de permitir el procesamiento en paralelo es usar WPP dentro de un sector regular, que se compararía con mosaicos, una "fila" de mosaicos incluidos en un solo sector. Los datos dentro de este sector, podrían codificarse también en paralelo utilizando subflujos de WPP, dentro del sector. La separación de instantánea en sectores 900 y 60 mosaicos/subflujos de WPP 914 se muestra en la figura 3/1 de los ejemplos.

- La figura 3, muestra por lo tanto la asignación de divisiones codificadas en paralelo tales como 906 o 914 a un sector o segmento de transporte de red (un solo paquete de red o múltiples paquetes de red 900).

La encapsulación de los datos codificados a unidades de capas de abstracción de red (NAL), como se define en H.264

o HEVC, antes de la transmisión o durante el proceso de codificación, agrega algo de encabezamiento a los bloques de datos, lo que permite la identificación de cada bloque y el reordenamiento de bloques, si es aplicable. En el caso estándar, no se requiere ninguna señalización adicional, puesto que el orden de los elementos de codificación se proporciona siempre en orden de decodificación, esto es una asignación implícita de la posición del mosaico o fragmento de codificación general.

Si el procesamiento en paralelo se considera con una capa de adicional para el transporte en paralelo de bajo retardo, esto es, la capa de transporte puede reordenar divisiones de instantánea para los mosaicos con el fin de permitir la transmisión de bajo retardo, lo que significa enviar fragmentos como se muestra en la figura 4 a medida que se codifican. Aquellos fragmentos pueden ser también sectores no completamente codificados, pueden ser un subconjunto de un sector o pueden estar contenidos en un sector dependiente.

En el caso de crear fragmentos adicionales, hay una solución intermedia entre eficacia, que sería la más alta con bloques de datos grandes debido a que la información de encabezamiento está agregando un número constante de bytes y retardo, debido a que los bloques de datos grandes de los codificadores en paralelo necesitarían almacenarse en memoria intermedia antes de la transmisión. El retardo global puede reducirse, si la representación codificada de los mosaicos verticales 906 se separa en un número de fragmentos 916 que se transmiten tan pronto como un fragmento se codifica completamente. El tamaño de cada fragmento puede determinarse en términos de una región de imagen fija, tal como macrobloques, LCU o en términos de datos máximos como se muestra en la figura 4.

Por lo tanto, la figura 4 muestra una fragmentación genérica de un fotograma con enfoque de codificación mosaico para un retardo de extremo a extremo mínimo.

De manera similar, la figura 5 muestra una fragmentación de un fotograma con enfoque de codificación de WPP para un retardo de extremo a extremo mínimo.

La transmisión puede agregar retardo adicional, por ejemplo, si se aplica el procesamiento orientado a bloques adicional, tales como códigos de corrección de error hacia delante que incrementan la robustez de la transmisión. Además, la infraestructura de red (encaminadores, etc.) o el enlace físico puede agregar retardo, esto se conoce típicamente como latencia para una conexión. Además de latencia, la tasa de bits de transmisión determina el tiempo (retardo) para transferir los datos de la parte a a la parte b, en una conversación como se muestra en la figura 6 que usa servicios de video.

Si los bloques de datos codificados se transmiten en orden incorrecto, el retardo de reordenamiento tiene que considerarse. La decodificación puede iniciar tan pronto como una unidad de datos ha llegado, suponiendo que estén disponibles otras unidades de datos que han de decodificarse antes de esta.

En el caso de mosaicos, no hay dependencias entre mosaicos, por lo tanto un mosaico puede decodificarse inmediatamente. Si los fragmentos se han producido de un mosaico, tales como sectores separados por cada fragmento como se muestra en la figura 4, los fragmentos pueden transportarse directamente, tan pronto como se codifiquen respectivamente en sus LCU contenidas o se hayan codificado las CU.

El dispositivo de presentación ensambla las salidas de motores de decodificación en paralelo y reenvía la instantánea combinada línea a línea a la pantalla.

La pantalla no necesariamente agrega ningún retardo, sino que en la práctica puede hacer algo de procesamiento de intra-fotograma antes de que los datos de instantánea se muestren realmente. Esto depende de las elecciones de diseño por el fabricante de hardware.

Resumiendo, se puede influir en las etapas de codificación, encapsulación, transmisión y decodificación con el fin de obtener un retardo de extremo a extremo mínimo. Si se usa el procesamiento en paralelo, los mosaicos y fragmentación dentro de los mosaicos, el retardo total puede reducirse significativamente como se muestra en la figura 7, en comparación con una cadena de procesamiento usada típicamente que agrega retardo de aproximadamente un fotograma en cada una de estas etapas como se muestra en la figura 8.

En particular, mientras que la figura 7 muestra codificación, transmisión y decodificación para mosaicos con subconjuntos genéricos con retardo de extremo a extremo mínimo, la figura 8 ilustra un retardo de extremo a extremo obtenido comúnmente.

HEVC permite el uso de división de sector, división de mosaico y de la siguiente manera.

mosaico: un número entero de bloques de árbol que tienen lugar concurrentemente en una columna y una fila, ordenados consecutivamente en la exploración de trama de bloque de árboles del mosaico. La división de cada instantánea en mosaicos es una partición. Los mosaicos en una instantánea se ordenan consecutivamente en la

exploración de trama de mosaico de la instantánea. Aunque un sector contiene bloques de árbol que son consecutivos en la exploración de trama de bloques de árbol de un mosaico, estos bloques de árbol no son necesariamente consecutivos en exploración de trama de bloque de árbol de la instantánea.

5 sector: un número entero de bloques de árbol ordenados consecutivamente en la exploración de trama. La división de cada instantánea en sectores es una partición. Las direcciones de bloque de árbol se derivan de la primera dirección de bloque de árbol en un sector (como se representa en el encabezamiento de sector).

10 exploración de trama: un mapeo de un patrón bidimensional rectangular a un patrón unidimensional, de tal manera que las primeras entradas en el patrón unidimensional son de la primera fila superior del patrón bidimensional explorado de izquierda a derecha, seguido de manera similar por la segunda, tercera, etc., filas del patrón (yendo hacia abajo) cada una explorada de izquierda a derecha.

15 bloque de árbol: un bloque de NxN de muestras de luminancia y dos bloques correspondientes de muestras de crominancia de una instantánea que tiene tres matrices de muestra o un bloque de NxN de muestras de una instantánea monocromática o una instantánea que se codifica utilizando tres planos de color separados. La división de un sector en bloques de árbol es una partición.

20 partición: la división de un conjunto en subconjuntos, de tal manera que cada elemento del conjunto está en exactamente uno de los subconjuntos.

árbol cuádruple: un árbol en el cual un nodo padre puede dividirse en cuatro nodos hijos. Un nodo hijo se puede convertir en nodo padre para otra división en cuatro nodos hijo.

25 A continuación, se explica la subdivisión espacial de instantáneas, sectores y mosaicos. En particular, la siguiente descripción específica cómo una instantánea se divide en sectores, mosaicos y bloques de árbol de codificación. Las instantáneas se dividen en sectores y mosaicos. Un sector es una secuencia de bloques de árbol de codificación. Asimismo, un mosaico es una secuencia de bloques de árbol de codificación.

30 Las muestras se procesan en unidades de bloques de árbol de codificación. El tamaño de matriz de luminancia para cada bloque de árbol en muestras tanto en anchura como en altura es CtbSize. La anchura y la altura de las matrices de crominancia para cada bloque de árbol de codificación son CtbWidthC y CtbHeightC, respectivamente. Por ejemplo, una instantánea puede dividirse en dos sectores como se muestra en la siguiente figura. Como otro ejemplo, una instantánea puede dividirse en tres mosaicos como se muestra en la segunda figura siguiente.

35 A diferencia de los sectores, los mosaicos son siempre rectangulares y siempre contiene un número entero de bloques de árbol de codificación en la exploración de trama de bloque de árbol de codificación. Un mosaico puede consistir en bloques de árbol de codificación contenidos en más de un sector. De manera similar, un sector puede comprender bloques de árbol de codificación contenidos en más de un mosaico.

40 La figura 9 ilustra una instantánea 898 con bloques de árbol de codificación 918 de 11 por 9 que se divide en dos sectores 900a,b.

45 La figura 10 ilustra una instantánea con bloques de árbol de codificación 918 de 13 por 8 que se divide en tres mosaicos.

50 Cada bloque de árbol 918 de codificación 898 se asigna a una señalización de partición para identificar los tamaños de bloque para intra- o inter- predicción y para codificación de transformada. La partición es una partición de árbol cuádruple recursiva. La raíz del árbol cuádruple está asociada con el bloque de árbol de codificación. El árbol cuádruple se divide hasta que se alcanza una hoja, que se hace referencia como el bloque de codificación. El bloque de codificación es el nodo de raíz de dos árboles, el árbol de predicción y el árbol de transformada.

55 El árbol de predicción especifica la posición y tamaño de los bloques de predicción. Los bloques de predicción y datos de predicción asociados se denominan unidad de predicción.

La figura 11 muestra una sintaxis de RBSP de conjunto de parámetro de secuencia a modo de ejemplo.

60 El árbol de transformada especifica la posición y tamaño de bloques de transformada. Los bloques de transformada y datos de transformada asociados se denominan unidad de transformada.

La información de división para luminancia y crominancia es idéntica para el árbol de predicción y puede ser idéntica o no para el árbol de transformada.

El bloque de codificación, los datos de codificación asociados y las unidades de predicción y de transformada

asociadas forman conjuntamente una unidad de codificación.

Un proceso para una conversión de una dirección de bloque de árbol de codificación en orden de trama de bloque de árbol de codificación a orden de exploración de mosaico podría ser como sigue:

5

Los resultados de este proceso son

10

- una matriz CtbAddrTS[ctbAddrRS], con ctbAddrRS en el intervalo de 0 a PicHeightInCtbs * PicWidthInCtbs - 1, inclusive.
- una matriz TileId[ctbAddrTS], con ctbAddrTS en el intervalo de 0 a PicHeightInCtbs * PicWidthInCtbs - 1, inclusive. La matriz CtbAddrTS[] se deriva como sigue:

15

```
para(ctbAddrRS = 0; ctbAddrRS < PicHeightInCtbs * PicWidthInCtbs, ctbAddrRS++) {
    tbX = ctbAddrRS % PicWidthInCtbs
    tbY = CtbAddrRS / PicWidthInCtbs
    para(j = 0; j <= num_tile_columns_minus 1; j++)
        si(tbX < ColBd[j + 1])
            tileX = j
    para(i = 0; i <= num_tile_rows_minus1: i++)
        si(tbY < RowBd[i + 1])
            tileY = i
    CtbAddrTS[ctbAddrRS] = ctbAddrRS - tbX
    para(i = 0; i < tileX; i++)
        ctbAddrTS += RowHeight[tileY] * ColumnWidth[i]
    CtbAddrTS[ctbAddrRS] += (tbY - RowBd[tileY])
    * ColumnWidth[tileY] + tbX - ColBd[tileX]
}
```

30

La matriz TileId[] se deriva como sigue:

35

```
para(j = 0, tileId = 0; j <= num_tile_columns_minus1; j++)
    para(i = 0; i <= num_tile_rows_minus 1; i++, tileId++)
        para(y = RowBd[j]; y < RowBd[j + 1]; y++)
            para(x = ColBd[i]; x < ColBd[i + 1]; x++)
                TileId[CtbAddrTS[y*PicWidthInCtbs + x]] = tileId
```

40

Una sintaxis a modo de ejemplo correspondiente se muestra en las figuras 11, 12 y 13, en la que la figura 12 tiene una sintaxis de RBSP de conjunto de parámetro de instantánea a modo de ejemplo. La figura 13 muestra una sintaxis de encabezamiento de sector a modo de ejemplo.

En el ejemplo de sintaxis, se puede aplicar la siguiente semántica:

45

entropy_slice_flag igual a 1 especifica que el valor de los elementos de sintaxis de encabezamiento de sector no presente se infiere para que sea igual al valor de elementos de sintaxis de encabezamiento de sector en un sector precedente, donde un sector precedente se define como el sector que contiene el bloque de árbol de codificación con ubicación (SliceCtbAddrRS - 1). entropy_slice_flag debe ser igual a 0 cuando SliceCtbAddrRS es igual a 0.

50

tiles_or_entropy_coding_sync_idc igual a 0 especifica que hay solamente un mosaico en cada instantánea en la secuencia de video codificada y ningún proceso de sincronización específico para variables de contexto se invoca antes de la decodificación del primer bloque de árbol de codificación de una fila de bloques de árbol de codificación.

55

tiles_or_entropy_coding_sync_idc igual a 1 especifica que puede haber más de un mosaico en cada instantánea en la secuencia de video codificada y ningún proceso de sincronización específico para variables de contexto se invoca antes de la decodificación del primer bloque de árbol de codificación de una fila de bloques de árbol de codificación.

60

tiles_or_entropy_coding_sync_idc igual a 2 especifica que hay solamente un mosaico en cada instantánea en la secuencia de video codificada, un proceso de sincronización específico para variables de contexto se invoca antes de la decodificación del primer bloque de árbol de codificación de una fila de bloques de árbol de codificación y un proceso de memorización específico para variables de contexto se invoca después de la decodificación de dos bloques de árbol de codificación de una fila de bloques de árbol de codificación.

65

El valor de tiles_or_entropy_coding_sync_idc estará en el intervalo de 0 a 2, inclusive.

num_tile_columns_minus1 más 1 especifica el número de columnas de mosaico que dividen la instantánea.

num_tile_rows_minus1 más 1 especifica el número de filas de mosaico que dividen la instantánea. Cuando num_tile_columns_minus1 es igual a 0, num_tile_rows_minus1 no será igual a 0.

5

Una o ambas de las siguientes condiciones serán satisfechas para cada sector y mosaico:

- Todos los bloques codificados en un sector pertenecen al mismo mosaico.

10

- Todos los bloques codificados en un mosaico pertenecen al mismo sector.

NOTA - Dentro de la misma instantánea, puede haber tanto sectores que contienen múltiples mosaicos como mosaicos que contienen múltiples sectores.

15

uniform_spacing_flag igual a 1 especifica que los límites de columna y asimismo límites de fila están distribuidos uniformemente a través de la instantánea. uniform_spacing_flag igual a 0 especifica que los límites de columna y asimismo los límites de fila no están distribuidos uniformemente a través de la instantánea sino señalizados explícitamente utilizando los elementos de sintaxis column_width[i] y row_height[i].

20

column_width[i] especifica la anchura de la i-ésima columna de mosaicos en unidades de bloques de árbol de codificación.

25

row_height[i] especifica la altura de la i-ésima fila de mosaicos en unidades de bloques de árbol de codificación. Los valores de [ColumnWidth[i], que especifican la anchura de la i-ésima columna de mosaicos en unidades de bloques de árbol de codificación y los valores de [ColumnWidthInLumaSamples[i], que especifican la anchura de la i-ésima columna de mosaicos en unidades de muestras de luminancia, se derivan como sigue:

30

```
para(i = 0; i <= num_tile_columns_minus1; i++) {
    si(uniform_spacing_flag)
        ColumnWidth[i] = ((i + 1) * PicWtdthInCtbs) / (num_tile_columns_minus1 + 1) -
            (i * PicWidthInCtbs) (num_tile_columns_minus1 + 1)
    si no
        ColumnWidth[i] = column_width[i]
        ColumnWidthInLumaSamples[i] = ColumnWidth[i] << Log2CtbSize }
```

35

Los valores de RowHeight[i], que especifican la altura de la i-ésima fila de mosaicos en unidades de bloques de árbol de codificación, se derivan como sigue:

40

```
para(i = 0; i <= num_tile_rows_minus1; i++)
    si(uniform_spacing_flag)
        RowHeight[i] = ((i + 1) * PicHeightInCtbs) / (num_tile_rows_minus1 + 1) -
            (i * PicHeightInCtbs) / (num_tile_rows_minus1 + 1)
    si no
        RowHeight[i] = row_height[i]
```

45

Los valores de ColBd[i], que especifican la ubicación del límite de columna izquierda de la i-ésima columna de mosaicos en unidades de bloques de árbol de codificación, se derivan como sigue:

50

```
para(ColBd[0] = 0, i = 0; i <= num_tile_columns_minus1; i++)
    ColBd[i + 1] = ColBd[i] + ColumnWidth[i]
```

Los valores de RowBd[i], que especifican la ubicación del límite de fila superior de la i-ésima fila de mosaicos en unidades de bloques de árbol de codificación, se derivan como sigue:

55

```
para(RowBd[0] = 0, i = 0; i <= num_tile_rows_minus1; i++)
    RowBd[i + 1] = RowBd[i] + RowHeight[i]
```

60

num_substreams_minus1 más 1 especifica el número máximo de subconjuntos incluidos en un sector cuando tiles_or_entropy_coding_sync_idc es igual a 2. Cuando no está presente, se infiere que el valor de num_substreams_minus1 es igual a 0.

num_entry_point_offsets especifica el número de elementos de sintaxis de entry_point_offset[i] en el encabezamiento de sector. Cuando tiles_or_entropy_coding_sync_idc es igual a 1, el valor de num_entry_point_offsets estará en el intervalo de 0 a (num_tile_columns_minus1 + 1) * (num_tile_rows_minus1 + 1)

- 1, inclusive. Cuando `tiles_or_entropy_coding_sync_idc` es igual a 2, el valor de `num_entry_point_offsets` estará en el intervalo de 0 a `num_substreams_minus1`, inclusive. Cuando no está presente, se infiere que el valor de `num_entry_point_offsets` es igual a 0.

5 `offset_len_minus1` más 1 especifica la longitud, en bits, de los elementos de sintaxis de `entry_point_offset[i]`.

`entry_point_offset[i]` especifica el desplazamiento del punto de entrada *i*-ésima, en bytes y se representará por `offset_len_minus1` más 1 bit. La unidad de NAL de sector codificada consiste en `num_entry_point_offsets` + 1 subconjuntos, con valores de índice de subconjunto que varían de 0 a `num_entry_point_offsets`, inclusive. El subconjunto 0 consiste en los bytes 0 a `entry_point_offset[0]` - 1, inclusive, de la unidad de NAL de sector codificada, subconjunto *k*, con *k* en el intervalo de 1 a `num_entry_point_offsets` - 1, inclusive, consiste en los bytes `entry_point_offset[k - 1]` a `entry_point_offset[k]` + `entry_point_offset[k - 1]` - 1, inclusive, de la unidad de NAL de sector codificada y el último subconjunto (con el índice de subconjunto igual a `num_entry_point_offsets`) consiste en los bytes restantes de la unidad de NAL de sector codificada.

15 NOTA - El encabezamiento de la unidad NAL y el encabezamiento de sector de una unidad de NAL de sector codificada están siempre incluidos en el subconjunto 0.

20 Cuando `tiles_or_entropy_coding_sync_idc` es igual a 1 y `num_entry_point_offsets` es mayor de 0, cada subconjunto contendrá todos los bits codificados de uno o múltiples mosaicos completos y el número de subconjuntos será igual o menor que el número de mosaicos en el sector. Cuando `tiles_or_entropy_coding_sync_idc` es igual a 2 y `num_entry_point_offsets` es mayor de 0, el subconjunto *k* para cada uno de todos los valores de *k* posibles contendrá todos los bits a usarse durante el proceso de inicialización para el puntero de flujo de bits actual *k*.

25 Con respecto a la semántica de datos de sector, lo siguiente se puede aplicar.

`end_of_slice_flag` igual a 0 especifica que otro macrobloque sigue en el sector. `end_of_slice_flag` igual a 1 especifica el fin del sector y que no sigue ningún macrobloque adicional.

30 `entry_point_marker_two_3bytes` es una secuencia de valor de fijo de 3 bytes igual a 0x000002. Este elemento de sintaxis se denomina prefijo de marcador de entrada.

`tile_idx_minus_1` especifica el `TileID` en orden de exploración de trama. El primer mosaico en la instantánea tendrá un `TileID` de 0. El valor de `tile_idx_minus_1` estará en el intervalo de 0 a $(\text{num_tile_columns_minus1} + 1) * (\text{num_tile_rows_minus1} + 1) - 1$.

Un proceso de análisis sintáctico de CABAC para los datos de sector podría ser como sigue:

40 Este proceso se invoca cuando se analizan sintácticamente elementos de sintaxis con el descriptor `ae(v)`.

Las entradas a este proceso son una petición para un valor de un elemento de sintaxis y valores de elementos de sintaxis analizados sintácticamente de manera previa.

45 La salida de este proceso es el valor del elemento de sintaxis.

50 Cuando se inicia el análisis sintáctico de los datos de sector de un sector, se invoca el proceso de inicialización de un proceso de análisis sintáctico de CABAC. Cuando `tiles_or_entropy_coding_sync_idc` es igual a 2 y `num_substreams_minus1` es mayor que 0, una tabla de mapeo `BitStreamTable` con `num_substreams_minus1` + 1 entradas especifica una tabla de puntero de flujo de bits para uso para derivación de puntero de flujo de bits actual posterior se deriva como sigue.

- `BitStreamTable[0]` es inicializada para contener el puntero de flujo de bits.

55 - Para todos los índices *i* mayores de 0 y menores que `num_substreams_minus1` + 1, `BitStreamTable[i]` contiene un puntero de flujo de bits a `entry_point_offset[i]` bytes después de `BitStreamTable[i - 1]`.

El puntero de flujo de bits actual se establece a `BitStreamTable[0]`.

60 La dirección del bloque de codificación mínimo del bloque de árbol de codificación que contiene el bloque vecino espacial `T`, `ctbMinCbAddrT`, se deriva utilizando una ubicación (*x0*, *y0*) de la muestra de luminancia superior izquierda del bloque de árbol de codificación actual, de tal manera que, por ejemplo, como sigue.

$$x = x_0 + 2 \ll \text{Log2CtbSize} - 1$$

$$y = y0 - 1$$

$$ctbMinCbAddrT = MinCbAddrZS[x \gg \text{Log2MinCbSize}] [y \gg \text{Log2MinCbSize}]$$

- 5 La variable availableFlagT se obtiene al invocar un proceso de derivación de disponibilidad de bloque de codificación apropiado con ctbMinCbAddrT como entrada.

Cuando se inicia el análisis sintáctico de un árbol de codificación y tiles_or_entropy_coding_sync_idc es igual a 2 y num_substreams_minus 1 es mayor de 0, se aplica lo siguiente.

- 10 - Si CtbAddrRS % PicWidthInCtbs es igual a 0, se aplica lo siguiente.
- Cuando availableFlagT es igual a 1, el proceso de sincronización del proceso de análisis sintáctico de CABAC se invoca como se especifica en el artículo subordinado "proceso de sincronización para variables de contexto".
 - 15 - Se invoca un proceso de decodificación para decisiones binarias antes de la terminación, seguido por un proceso de inicialización para el motor de decodificación aritmética.
 - El puntero de flujo de bits actual se ajusta para indicar la BitStreamTable[i] con el índice i derivado como sigue.

$$i = (\text{CtbAddrRS} / \text{PicWidthInCtbs}) \% (\text{num_substreams_minus1} + 1)$$

- 25 - De otra manera, si CtbAddrRS % PicWidthInCtbs es igual a 2, el proceso de memorización del proceso de análisis sintáctico de CABAC se invoca como se especifica en el artículo subordinado "proceso de memorización para variables de contexto".

Un proceso de inicialización podría ser como sigue:

- 30 Las salidas de proceso son variables internas de CABAC inicializadas.

Se invocan procesos especiales del mismo cuando se inicia el análisis sintáctico de los datos de sector de un sector o cuando se inicia el análisis sintáctico de los datos de un árbol de codificación y el árbol de codificación es el primer árbol de codificación en un mosaico.

- 35 El proceso de memorización para variables de contexto podría ser como sigue:

Las entradas de este proceso son las variables de contexto de CABAC indexadas por ctxIdx.

- 40 La salida de este proceso son las variables TableStateSync y TableMPSSync que contienen los valores de las variables n y m usadas en el proceso de inicialización de variables de contexto que se asignan a elementos de sintaxis excepto para la bandera de fin de sector.

- 45 Para cada variable de contexto, las entradas correspondientes n y m de las tablas TableStateSync y TableMPSSync se inicializan a los pStateldx y valMPS correspondientes.

El proceso de sincronización para variables de contexto podría ser como sigue:

- 50 Las entradas de este proceso son las variables TableStateSync y TableMPSSync que contienen los valores de las variables n y m usadas en el proceso de memorización de variables de contexto que se asignan a elementos de sintaxis excepto para la bandera de fin de sector.

Las salidas de este proceso son las variables de contexto de CABAC indexadas por ctxIdx.

- 55 Para cada variable de contexto, las variables de contexto correspondientes pStateldx y valMPS se inicializan a las entradas correspondientes n y m de las tablas TableStateSync y TableMPSSync.

- 60 A continuación, se explica una codificación y transporte de bajo retardo utilizando WPP. En particular, el análisis a continuación revela en cuanto a cómo el transporte de bajo retardo como se describe en la figura 7 puede también aplicarse a WPP.

Antes de nada, es importante que un subconjunto de la instantánea pueda enviarse, antes de la finalización de toda la instantánea. Normalmente, esto es obtenible utilizando sectores, como ya se muestra en la figura 5.

- Con el fin de reducir el retardo en comparación con mosaicos, como se muestra en las siguientes figuras, se necesita aplicar un solo subflujo de WPP por fila de LCU y permitir además la transmisión separada de cada una de estas filas. Con el fin de mantener la eficacia de codificación alta, los sectores por cada fila/subflujo no pueden usarse. Por consiguiente, a continuación, se presenta un denominado sector dependiente como se define en la sección siguiente.
- 5 Este sector, por ejemplo, no tiene todos los campos del encabezamiento de sector de HEVC completo, sino los campos usados para sectores de entropía. Además, puede haber un conmutador para desconectar la interrupción de CABAC entre filas. En el caso de WPP, el uso del contexto de CABAC (flechas en la figura 14) y predicción de filas se permitirá para mantener la ganancia de eficacia de codificación de WPP sobre los mosaicos.
- 10 En particular, la figura 14 ejemplifica una instantánea 10 para WPP en un sector 900 regular (reg. SL), y, para procesamiento de bajo retardo, en sectores dependientes (OS) 920.
- Actualmente el estándar de HEVC próximo ofrece dos tipos de partición en términos de sector. Hay sectores regulares (normales) y sectores de entropía. El sector regular es completamente independiente de la partición de instantánea excepto que algunas dependencias pueden estar disponibles debido al desbloqueo de proceso de filtro en límites de sector. El sector de entropía es también independiente pero solamente en términos de codificación de entropía. La idea de la figura 14 es generalizar el concepto de sectorización. Por lo tanto, el estándar de HEVC próximo debe ofrecer dos tipos generales de sectores: independientes (regulares) o dependientes. Por consiguiente, se presenta un nuevo tipo de sector, un sector dependiente.
- 15
- 20 El sector dependiente es un sector que tiene dependencias al sector previo. Las dependencias son un dato particular que puede utilizarse entre sector en el proceso de decodificación de entropía y/o proceso de reconstrucción de pixel.
- En la figura 14, se presenta a modo de ejemplo un concepto de sectores dependientes. La instantánea se inicia, por ejemplo, siempre con un sector regular. Obsérvese que en este concepto, el comportamiento de sector regular cambia ligeramente. Típicamente, en los estándares como H264/AVC o HEVC, el sector regular es completamente independiente de la partición y no tiene que mantener ningún dato después de la decodificación excepto algunos datos para el desbloqueo del proceso de filtro. Sin embargo, el procesamiento de sector dependiente próximo 920 es solamente posible haciendo referencia a los datos del sector anterior, en este punto en la primera fila: sector regular 900. Para establecer esto, los sectores regulares 900 deben mantener los datos de la última fila de CU. Estos datos comprenden:
- 25
- 30
- datos de motor de codificación de CABAC (estados de modelo de contexto de una CU desde la que puede iniciarse el proceso de decodificación de entropía del sector dependiente),
 - todos los elementos de sintaxis decodificados de CU para el proceso de decodificación de CABAC regular de los CU dependientes,
 - los datos de predicción intra y de vector de movimiento.
- 35
- 40
- En consecuencia, cada sector dependiente 920 hará el mismo procedimiento, mantener los datos para sector dependiente próximo de fila en la misma instantánea.
- En la práctica, estas etapas adicionales no deberían ser un problema, debido a que el proceso de decodificación en general se ve forzado siempre a almacenar algunos datos como elementos de sintaxis.
- 45
- En las secciones a continuación se presentan posibles cambios para la sintaxis del estándar de HEVC que se requieren para posibilitar el concepto de sectores dependientes.
- 50 La figura 5, por ejemplo, ilustra cambios posibles en la sintaxis de RBSP del conjunto de parámetros de instantánea.
- Las semánticas del conjunto de parámetros de instantánea para sectores dependientes podrían ser como sigue:
- 55
- dependent_slices_present_flag** igual a 1 especifica que la instantánea está conteniendo sectores dependientes y el proceso de decodificación de cada sector (regular o dependiente) almacenará estados de decodificación de entropía y datos de predicción de intra y vector de movimiento para el siguiente sector que puede ser un sector dependiente que puede también seguir al sector regular. El siguiente sector dependiente puede hacer referencia a esos datos almacenados.
- 60 La figura 16 muestra una sintaxis de slice_header posible con los cambios relativos al estado actual de HEVC.
- dependent_slice_flag** igual a 1 especifica que el valor de los elementos de sintaxis de encabezamiento de sector no presente se infiere para que sea igual al valor de los elementos de sintaxis de encabezamiento de sector en un sector precedente (regular), en donde un sector precedente se define como el sector que contiene el bloque de

árbol de codificación con ubicación ($\text{SliceCtbAddrRS} - 1$). $\text{dependent_slice_flag}$ será igual a 0 cuando SliceCtbAddrRS es igual a 0.

no_cabac_reset_flag igual a 1 especifica la inicialización de CABAC del estado guardado del sector previamente decodificado (y no con valores iniciales). De otra manera, esto es, si es igual a 0, la inicialización de CABAC independiente de cualquier estado del sector decodificado previamente, esto es, con valores iniciales.

last_ctb_cabac_inic_flag igual a 1 especifica la inicialización de CABAC del estado guardado del último bloque de árbol codificado del sector previamente decodificado (por ejemplo, para mosaicos siempre igual a 1). De otra manera (igual a 0), los datos de inicialización se hacen referencia desde el estado guardado del segundo bloque de árbol codificado de la última fila ctb (vecina) del sector previamente decodificado, si el primer bloque de árbol codificado del sector actual es el primer bloque de árbol codificado en la fila (esto es, modo WPP), de otra manera se efectúa la inicialización de CABAC del estado guardado del último bloque de árbol codificado del sector decodificado previamente.

Se proporciona a continuación una comparación de sectores dependientes y otros esquemas de división (informativa).

En la figura 17, se muestra la diferencia entre sectores normales y dependientes.

Una codificación y transmisión posible de subflujos de WPP en sectores dependientes (DS) como se ilustra con respecto a la figura 18 compara una codificación para el transporte de bajo retardo de mosaicos (izquierda) y WPP/DS (derecha). Las cruces en negrita dibujadas de manera continua en la figura 18 muestran el mismo punto de tiempo para los dos métodos suponiendo que la codificación de la fila de WPP toma el mismo tiempo que la codificación de un solo mosaico. Debido a las dependencias de codificación, solamente la primera línea de WPP está preparada, después que todos los mosaicos se han codificado. Sin embargo, utilizando el enfoque de sector dependiente permite que el enfoque de WPP envíe la primera fila una vez que esta codificada. Esto es diferente de asignaciones de subflujo de WPP anteriores, "subflujo" se define para WPP como una concatenación de filas de CU de sector a decodificarse por WPP por el mismo hilo de decodificador, esto es, el mismo núcleo/procesador. Aunque un subflujo por fila y por sector de entropía también habría sido posible antes, el sector de entropía rompe las dependencias de codificación de entropía y tiene por consiguiente eficacia de codificación más baja, esto es, la ganancia de eficacia de WPP se pierde.

Adicionalmente, la diferencia de retardo entre ambos enfoques puede ser realmente baja, suponiendo una transmisión como se muestra en la figura 19. En particular, la figura 19 ilustra una codificación de WPP con transmisión de bajo retardo en tubería.

Suponiendo que la codificación de las últimas dos CU de DS n.º 1.1 en el enfoque de WPP en la figura 18, no toma más tiempo que la transmisión de la primera fila SL n.º 1, no hay ninguna diferencia entre los mosaicos y WPP en el caso de bajo retardo. Sin embargo, la eficacia de codificación de WP/DS supera el concepto de mosaico.

Con el fin de incrementar la robustez del modo de bajo retardo de WPP, la figura 20 ilustra que se obtiene robustez mejorada al usar sectores regulares (RS) como anclas. En la instantánea mostrada en la figura 20 un sector (regular (RS)) es seguido por sectores dependientes (DS). En este punto, el sector (regular) actúa como un ancla para romper las dependencias a sectores precedentes, por lo tanto, se proporciona más robustez en tal punto de inserción de un sector (regular). En principio, esto no es diferente de insertar sectores (regulares) de cualquier manera.

El concepto de sectores dependientes podría también implementarse como sigue.

En este punto, la figura 21 muestra una sintaxis de encabezamiento de sector posible.

La semántica de encabezamientos de sector es como sigue:

dependent_slice_flag igual a 1 especifica que el valor de cada elemento de sintaxis de encabezamiento de sector no presente se infiere para que sea igual al valor del elemento de sintaxis de encabezamiento de sector correspondiente en el sector precedente que contiene el bloque de árbol de codificación para el cual la dirección de bloque de árbol de codificación es $\text{SliceCtbAddrRS} - 1$. Cuando no está presente, se infiere que el valor de $\text{dependent_slice_flag}$ es igual a 0. El valor de $\text{dependent_slice_flag}$ será igual a 0 cuando SliceCtbAddrRS es igual a 0.

slice_address especifica la dirección en resolución de granularidad de sector en la cual el sector se inicia. La longitud del elemento de sintaxis slice_address es de $(\text{Ceil}(\text{Log}_2(\text{PicWidthInCtbs} * \text{PicHeightInCtbs})) + \text{SliceGranularity})$ bits.

La variable SliceCtbAddrRS , que especifica el bloque de árbol de codificación en el cual el sector se inicia en el orden de exploración de trama de bloques de árbol de codificación, se deriva como sigue.

`SliceCtbAddrRS = (slice_address >> SliceGranularity`

- 5 La variable SliceCbAddrZS, que especifica la dirección del primer bloque de codificación en el sector en granularidad del bloque de codificación mínimo en orden de exploración en z, se deriva como sigue.

`SliceCbAddrZS = slice_address
<< ((log2_diff_max_min_coding_block_size - SliceGranularity) <<1)`

- 10 La decodificación de sector inicia con la unidad de codificación más grande posible o en otros términos, CTU, en la coordenada de inicio de sector.

- 15 **first_slice_in_pic_flag** indica si el sector es el primer sector de la instantánea. Si **first_slice_in_pic_flag** es igual a 1, las variables SliceCbAddrZS y SliceCtbAddrRS se ajustan ambas a 0 y la decodificación se inicia con el primer bloque de árbol de codificación en la instantánea.

pic_parameter_set_id especifica el conjunto de parámetros de instantánea en uso. El valor de **pic_parameter_set_id** estará en el intervalo de 0 a 255, inclusive.

- 20 **num_entry_point_offsets** especifica el número de elementos de sintaxis de **entry_point_offset[i]** en el encabezamiento de sector. Cuando **tiles_or_entropy_coding_sync_idc** es igual a 1, el valor de **num_entry_point_offsets** estará en el intervalo de 0 a $(\text{num_tile_columns_minus_1} + 1) * (\text{num_tile_rows_minus_1} + 1) - 1$, inclusive. Cuando **tiles_or_entropy_coding_sync_idc** es igual a 2, el valor de **num_entry_point_offsets** estará en el intervalo de 0 a $\text{PicHeightInCtbs} - 1$, inclusive. Cuando no está presente, el valor de **num_entry_point_offsets** se infiere para que sea igual a 0.

offset_len_minus1 más 1 especifica la longitud, en bits, de los elementos de sintaxis de **entry_point_offset[1]**.

- 30 **entry_point_offset[i]** especifica el i-ésimo desplazamiento de punto de entrada, en bytes y se representará por **offset_len_minus1** más 1 bit. Los datos de sector codificados después del encabezamiento de sector consisten en los subconjuntos de **num_entry_point_offsets** + 1 con los valores del índice de subconjuntos que varían de 0 a **num_entry_point_offsets**, inclusive. El subconjunto 0 consiste en los bytes 0 a **entry_point_offset[0]**-1, inclusive, de los datos de sector codificados, subconjunto k, con k el intervalo de 1 a **num_entry_point_offsets** -1, inclusive, consiste en **entry_point_offset[k - 1]** bytes a **entry_point_offset[k] + entry_point_offset[k - 1]** -1, inclusive, de los datos de sector codificados y el último subconjunto (con el índice de subconjunto igual a **num_entry_point_offsets**) consiste en los bytes restantes de los datos de sector codificados.

- 40 Cuando **tiles_or_entropy_coding_sync_idc** es igual a 1 y **num_entry_point_offsets** es mayor de 0, cada subconjunto contendrá todos los bits codificados de exactamente un mosaico y el número de subconjuntos (esto es, el valor de **num_entry_point_offsets** + 1) será igual o menor que el número de mosaicos en el sector.

NOTA - Cuando **tiles_or_entropy_coding_sync_idc** es igual a 1, cada sector debe incluir ya sea un subconjunto de un mosaico (en tal caso la señalización de puntos de entrada es innecesaria) o un número entero de mosaicos completos.

- 45 Cuando **tile_or_entropy_coding_sync_idc** es igual a 2 y **num_entry_point_offsets** es mayor de 0, cada subconjunto k con k en intervalo de 0 a **num_entry_point_offsets** - 1, inclusive, contendrá todos los bits codificados de exactamente una fila de bloques de árbol de codificación, el último subconjunto (con el índice de subconjunto igual a **num_entry_point_offsets**) contendrá todos los bits codificados de los bloques de codificación restantes incluidos en el sector, en el que los bloques de codificación restantes consisten ya sea en exactamente una fila de bloques de árbol de codificación o un subconjunto de una fila de bloques de árbol de codificación y el número de subconjuntos (esto es, el valor de **num_entry_point_offsets** + 1) será igual al número de filas de bloques de árbol de codificación en el sector, en el que un conjunto de una fila de bloques de árbol de codificación en el sector también se cuenta.

- 55 NOTA - Cuando **tiles_or_entropy_coding_sync_idc** es igual a 2, un sector puede incluir un número de filas de bloques de árbol de codificación y un subconjunto de una fila de bloques de árbol de codificación. Por ejemplo, si un sector incluye dos filas y media de bloques de árbol de codificación, el número de subconjuntos (esto es, el valor de **num_entry_point_offsets** + 1) será igual a 3.

- 60 La sintaxis de RBSP del conjunto de parámetros de instantánea correspondiente podría seleccionarse como se ilustra en la figura 22.

La semántica de RBSP del conjunto de parámetros de instantánea podría ser como sigue:

- 65 **dependent_slice_enabled_flag** igual a 1 especifica la presencia del elemento de sintaxis **dependent_slice_flag** en el encabezamiento de sector para instantáneas codificadas que hacen referencia al conjunto de parámetros de

instantánea. `dependent_slice_enabled_flag` igual a 0 especifica la ausencia del elemento de sintaxis `dependent_slice_flag` en el encabezamiento de sector para instantáneas codificadas que hacen referencia al conjunto de parámetro de instantánea. Cuando `tiles_or_entropy_coding_sync_idc` es igual a 3, el valor de `dependent_slice_enabled_flag` será igual a 1.

5

`tiles_or_entropy_coding_sync_idc` igual a 0 especifica que habrá solamente un mosaico en cada instantánea que hace referencia al conjunto de parámetros de instantánea, no habrá ningún proceso de sincronización específico para variables de contexto invocadas antes de la decodificación del primer bloque de árbol de codificación de una fila de bloques de árbol de codificación en cada instantánea que hace referencia al conjunto de parámetros de instantánea y los valores de `cabac_independent_flag` y `dependent_slice_flag` para instantáneas codificadas que hacen referencia al conjunto de parámetro de instantánea no serán ambas iguales a 1.

10

Nota, cuando `cabac_independent_flag` y `dependent_slice_flag` son ambos igual a 1 para un sector, el sector es un sector de entropía.

15

`tiles_or_entropy_coding_sync_idc` igual a 1 especifica que puede haber más de un mosaico en cada instantánea que hace referencia al conjunto de parámetros de instantánea, no habrá ningún proceso de sincronización específico para variables de contexto invocadas antes de la decodificación del primer bloque de árbol de codificación de una fila de bloques de árbol de codificación en cada instantánea que hace referencia al conjunto de parámetros de instantánea y los valores de `cabac_independent_flag` y `dependent_slice_flag` para instantáneas codificadas que hacen referencia al conjunto de parámetros de instantánea no serán ambas iguales a 1.

20

`tiles_or_entropy_coding_sync_idc` igual a 2 especifica habrá solamente un mosaico en cada instantánea que hace referencia al conjunto de parámetros de instantánea, un proceso de sincronización específico para variables de contexto será invocado antes de la decodificación del primer bloque de árbol de codificación de una fila de bloques de árbol de codificación en cada instantánea que hace referencia al conjunto de parámetros de instantánea y un proceso de memorización específico para variables de contexto será invocado después de la decodificación de dos bloques de árbol de codificación de una fila de bloques de árbol de codificación en cada instantánea que hace referencia al conjunto de parámetros de instantánea y los valores de `cabac_independent_flag` y `dependent_slice_flag` para instantáneas codificadas que hacen referencia al conjunto de parámetros de instantánea no serán ambas igual a 1.

25

30

`tiles_or_entropy_coding_sync_idc` igual a 3 especifica que habrá solamente un mosaico en cada instantánea que hace referencia al conjunto de parámetros de instantánea, no habrá ningún proceso de sincronización específico para variables de contexto invocadas antes de la decodificación del primer bloque de árbol de codificación de una fila de bloques de árbol de codificación en cada instantánea que hace referencia al conjunto de parámetros de instantánea y los valores de `cabac_independent_flag` y `dependent_slice_flag` para instantáneas codificadas que hacen referencia al conjunto de parámetros de instantánea pueden ambas ser iguales a 1.

35

40 Cuando `dependent_slice_enabled_flag` sea igual a 0, `tiles_or_entropy_coding_sync_idc` no será igual a 3.

Es un requisito de la conformación del flujo de bits que el valor de `tiles_or_entropy_coding_sync_idc` sea el mismo para todos los conjuntos de parámetro de instantánea que están activados dentro de una secuencia de video codificada.

45

Para cada sector que hace referencia al conjunto de parámetros de instantánea, cuando `tiles_or_entropy_coding_sync_idc` es igual a 2 y el primer bloque de codificación en el sector no es el primer bloque de codificación en el primer bloque de árbol de codificación de una fila de bloques de árbol de codificación, el último bloque de codificación en el sector pertenecerá a la misma fila de bloques de árbol de codificación como el primer bloque de codificación en el sector.

50

`num_tile_columns_minus1` más 1 especifica el número de columnas de mosaico que dividen la instantánea.

`num_tile_rows_minus1` más 1 especifica el número de filas de mosaico que dividen la instantánea.

55

Cuando `num_tile_columns_minus1` es igual a 0, `num_tile_rows_minus1` no será igual a 0. `Uniform_spacing_flag` igual a 1 especifica que los límites de columna y de manera análoga los límites de fila están distribuidos uniformemente a través de la instantánea. `Uniform_spacing_flag` igual a 0 especifica que los límites de columna y de manera análoga los límites de fila no están distribuidos uniformemente a través de la instantánea sino señalados explícitamente utilizando los elementos de sintaxis `column_width[i]` y `row_height[i]`.

60

`column_width[i]` especifica la anchura de la i-ésima columna de mosaico en unidades de bloques de árbol de codificación.

row_height[i] especifica la altura de la i-ésima fila de mosaico en unidades de bloques de árbol de codificación.

El vector colWidth[i] especifica la anchura de la i-ésima columna de mosaicos en unidades de CTB con la columna i que varía de 0 a num_tile_columns_minus 1, inclusive.

5

El vector CtbAddrRStoTS[ctbAddrRS] especifica la conversación de una dirección de CTB en orden de exploración de trama a una dirección de CTB en orden de exploración de mosaico con el índice ctbAddrRS que varía de 0 a (picHeightInCtbs * picWidthInCtbs) - 1, inclusive.

10

El vector CtbAddrTStoRS[ctbAddrTS] especifica la conversación de una dirección de CTB en orden de exploración de mosaico a una dirección de CTB en orden de exploración de trama con el índice ctbAddrTS que varía de 0 a (picHeightInCtbs * picWidthInCtbs) - 1, inclusive.

15

El vector TileId[ctbAddrTS] especifica la conversación de una dirección de CTB en orden de exploración de mosaico a una id de mosaico con ctbAddrTS que varía de 0 a (picHeightInCtbs * picWidthInCtbs) - 1, inclusive.

Los valores de colWidth, CtbAddrRStoTS, CtbAddrTStoRS y TileId se derivan al invocar un proceso de trama de CTB y conversación de exploración de mosaico con PicHeightInCtbs y PicWidthInCtbs como entradas y la salida se asigna a colWidth, CtbAddrRStoTS y TileId.

20

Los valores de ColumnWidthInLumaSamples[i], que especifican la anchura de la i-ésima columna de mosaicos en unidades de muestras de luminancia, se ajustan para que sean iguales a colWidth[i] << Log2CtbSize.

25

La matriz MinCbAddrZS[x][y], que especifica la conversación de una ubicación (x, y) en unidades de CB mínimas a una dirección de CB mínima en orden de exploración en z con x que varía de 0 a picWidthInMinCbs - 1, inclusive y que varía de 0 a picHeightInMinCbs - 1, inclusive, se deriva al invocar un proceso de inicialización de matriz de orden de exploración en z con Log2MinCbSize, Log2CtbSize, PicHeightInCtbs, PicWidthInCtbs y el vector CtbAddrRStoTS como entradas y la salida se asigna a MinCbAddrZS.

30

loop_filter_across_tiles_enabled_flag igual a 1 especifica que las operaciones de filtración en bucle se efectúan a través de límites de mosaico, **loop_filter_across_tile_enabled_flag** igual a 0 especifica que las operaciones de filtración en bucle no se efectúan a través de límites de mosaicos. Las operaciones de filtración en bucle incluyen el filtro de desbloqueo, desplazamiento adaptable de muestra y operaciones de filtro de bucle adaptables. Cuando no está presente, el valor de **loop_filter_across_tiles_enabled_flag** se infiere para que sea igual a 1.

35

cabac_independent_flag igual a 1 especifica que la decodificación de CABAC de bloques de codificación en un sector es independiente de cualquier estado del sector previamente decodificado. **cabac_independent_flag** igual a 0 especifica que la decodificación de CABAC de bloques de codificación en un sector es dependiente de los estados del sector previamente decodificado. Cuando no está presente, el valor de **cabac_independent_flag** se infiere para que sea igual a 0.

40

Un proceso de derivación para la disponibilidad de un bloque de codificación con una dirección de bloque de codificación mínima podría ser como sigue:

45

Las entradas a este proceso son

- una dirección de bloque de codificación mínimo CbAddrZS en orden de exploración en z

- la dirección de bloque de codificación mínima actual CBAddrZS en orden de exploración en z

50

La salida de este proceso es la disponibilidad del bloque de codificación con dirección de bloque de codificación mínima cbAddrZS en orden de exploración en z cbAvailable.

55

NOTA 1 - El significado de disponibilidad se determina cuando este proceso se invoca. NOTA 2 - cualquier bloque de codificación, independientemente de su tamaño, está asociado con una dirección de bloque de codificación mínima, que es la dirección del bloque de codificación con el tamaño de bloque de codificación mínimo en orden de exploración en z.

60

- Si una o más de las siguientes condiciones son ciertas, cbAvailable se ajusta a FALSO.

- minCbAddrZS es menor que 0

- minCbAddrZS es mayor que currMinCBAddrZS

- el bloque de codificación con dirección de bloque de codificación mínima minCbAddrZS pertenece a un sector diferente que el bloque de codificación con la dirección de bloque de codificación mínima actual currMinCbAddrZS y la dependent_slice_flag del sector que contiene el bloque de codificación con la dirección de bloque de codificación mínima actual currMinCbAddrZS es igual a 0.
 - el bloque de codificación con dirección de bloque de codificación mínima minCbAddrZS está contenido en un mosaico diferente que el bloque de codificación con la dirección de bloque de codificación mínima actual currMinCbAddrZS.
 - De otra manera, cbAvailable se ajusta a VERDADERO.
- Un proceso de análisis sintáctico de CABAC para datos de sector podría ser como sigue:
- Este proceso se invoca cuando se analizan sintácticamente ciertos elementos de sintaxis con el descriptor ae(v).
- Las entradas a este proceso son una petición para un valor de un elemento de sintaxis y valores de elementos de sintaxis analizados sintácticamente de manera previa.
- La salida de este proceso es el valor del elemento de sintaxis.
- Cuando se inicia el análisis sintáctico de los datos de un sector, se invoca el proceso de inicialización del proceso de análisis sintáctico de CABAC.
- La figura 23 ilustra cómo se busca un vecino espacial T para invocar el proceso de derivación de disponibilidad del bloque de árbol de codificación al bloque de árbol de codificación actual (informativo).
- La dirección de bloque de codificación mínimo del bloque de árbol de codificación que contiene el bloque vecino espacial T (figura 23), ctbMinCbAddrT, se deriva utilizando la ubicación (x0,y0) de la muestra de luminancia superior izquierda del bloque de árbol de codificación actual como sigue.
- $$x = x0 + 2 \ll \text{Log2CtbSize} - 1$$
- $$y = y0 - 1$$
- $$\text{ctbMinCbAddrT} = \text{MinCbAddrZS}[x \gg \text{Log2MinCbSize}] [y \gg \text{Log2MinCbSize}]$$
- La variable availableFlagT se obtiene al invocar un proceso de derivación de disponibilidad de bloque de codificación con ctbMinCbAddrT como entrada.
- Cuando se inicia el análisis sintáctico de un árbol de codificación como se especifica, se aplican las siguientes etapas ordenadas.
- El motor de decodificación aritmético se inicializa como sigue.
- Si CtbAddrRS es igual a slice_address, dependent_slice_flag es igual a 1 y entropy_coding_reset_flag es igual a 0, se aplica lo siguiente.
- El proceso de sincronización del proceso de análisis sintáctico de CABAC se invoca con TableStatIdxDS y TableMPSValDS como entrada.
- Un proceso de decodificación para decisiones binarias antes de la terminación se invoca, seguido por un proceso de inicialización para el motor de decodificación aritmético.
- De otra manera, si tiles_or_entropy_coding_sync_idc es igual a 2 y CtbAddrRS % PicWidthInCtbs es igual a 0, se aplica lo siguiente.
- Cuando availableFlagT es igual a 1, un proceso de sincronización del proceso de análisis sintáctico de CABAC se invoca con TableStatIdxWPP y TableMPSValWPP como entrada.
- Un proceso de decodificación para decisiones binarias antes de la terminación se invoca, seguido por el proceso para el motor de decodificación aritmética.
- Cuando cabac_independent_flag es igual a 0 y dependent_slice_flag es igual a 1 o cuando tiles_or_entropy_coding_sync_idc es igual a 2, el proceso de memorización se aplica como sigue.

- 5 Cuando `tiles_or_entropy_coding_sync_idc` es igual a 2 y `CtbAddrRS % PicWidthInCtbs` es igual a 2, el proceso de memorización del proceso de análisis sintáctico de CABAC se invoca con `TableStateldxWPP` y `TableMPSValWPP` como salida.
- 10 Cuando `cabac_independent_flag` es igual a 0, `dependent_slice_flag` es igual a 1 y `end_of_slice_flag` es igual a 1, el proceso de memorización del proceso de análisis sintáctico de CABAC se invoca con `TableStateldxDS` y `TableMPSValDS` como salida.
- 10 El análisis sintáctico de los elementos de sintaxis procede como sigue:
- Por cada valor requerido de un elemento de sintaxis se deriva una conversión a binario.
- 15 La conversión a binario para el elemento de sintaxis y la secuencia de binarios analizados sintácticamente determina el flujo del proceso de decodificación.
- Por cada binario de la conversión a binario del elemento de sintaxis, que se indexa por la variable `binIdx` se deriva un índice de contexto `ctxIdx`.
- 20 Por cada `ctxIdx` se invoca el proceso de decodificación aritmético.
- La secuencia resultante (`b0..bbinIdx`) de los binarios analizados sintácticamente se compara con el conjunto de series de binarios dadas por el proceso de conversión a binario después de la decodificación de cada binario. Cuando la secuencia coincide con una serie de binarios en el conjunto dado, el valor correspondiente se asigna al elemento de sintaxis.
- 25 En el caso de que la petición para un valor de un elemento de sintaxis se procese para el elemento de sintaxis `pcm_flag` y el valor decodificado de `pcm_flag` es igual a 1, el motor de decodificación se inicializa después de la decodificación de cualquier `pcm_alignment_zero_bit`, `num_subsequent_pcm` y todos los datos de `pcm_sample_luma` y `pcm_sample_chroma`.
- 30 Por lo tanto, la descripción anterior revela un decodificador como se muestra en la figura 24. Este decodificador, que se indica en general por el signo de referencia 5, reconstruye una instantánea 10 de un flujo de datos 12 al cual se codifica la instantánea 10 en unidades de sectores 14 a las cuales se divide la instantánea 10, en el que el decodificador 5 está configurado para decodificar los sectores 14 del flujo de datos 12 según un orden de sector 16. Naturalmente, el decodificador 5 no está restringido a decodificar los sectores 14. En su lugar, el decodificador 5 puede usar procesamiento en paralelo de frente de onda para decodificar los sectores 14, con la condición de que las instantáneas 10 que se dividen en sectores 14 sean apropiadas para procesamiento en paralelo de frente de onda. Por consiguiente, el decodificador 5 puede ser por ejemplo un decodificador que es apto para decodificar sectores 14 en paralelo de manera escalonada con el inicio de la decodificación de los sectores 14 teniendo en cuenta el orden de sector 16 para permitir el procesamiento de frente de onda como se ha descrito anteriormente y que se describirá posteriormente también.
- 35 El decodificador 5 responde a una porción de elemento de sintaxis 18 dentro de un sector actual de los sectores 14 para decodificar el sector actual según uno de al menos dos modos 20 y 22. Según uno primero de los al menos dos modos, es decir el modo 20, el sector actual se decodifica del flujo de datos 12 utilizando decodificación de entropía adaptable de contexto que incluye una derivación de límites de sector a través del contexto, es decir a través de las líneas discontinuas en la figura 24, esto es, al usar información que se deriva de la codificación/decodificación de otro "orden de sector 16 precedente a los sectores". Además, la decodificación del sector actual del flujo de datos 12 utilizando el primer modo 20 comprende una actualización continua de probabilidades del símbolo del códec y una inicialización de las probabilidades de símbolo al comienzo de la decodificación del sector actual, que depende de los estados guardados de las probabilidades de símbolo de un sector previamente decodificado. Tal dependencia se describió anteriormente, por ejemplo, en relación con el "proceso de sincronización para variables de códec". Finalmente, el primer modo 20 también involucra decodificación predictiva a través de los límites de sector. Tal decodificación predictiva a través de los límites de sector puede involucrar, por ejemplo, intra-predicción a través de los límites de sector, esto es, predicción de valores de muestra dentro del sector actual en base a los valores de muestra ya reconstruidos de "orden de sector 16", sector precedente o una predicción de parámetros de codificación a través de límites de sector tal como una predicción de vectores de movimiento, modos de predicción, modos de codificación o similares.
- 50
- 55
- 60 Según el segundo modo 22, el decodificador 5 decodifica el sector actual, esto es, el sector actualmente a decodificarse, del flujo de datos 12 utilizando decodificación de entropía adaptable al contexto con restricción, sin embargo, de la derivación de los contextos para no cruzar los límites de sector. Si por ejemplo, una plantilla de posiciones vecinas usada para derivar el contexto para un cierto elemento de sintaxis concerniente con un bloque

- dentro del sector actual se extiende a un sector vecino, cruzando de esta manera el límite de sector del sector actual, el atributo correspondiente de la porción respectiva del sector vecino, tal como el valor del elemento de sintaxis correspondiente de esta porción vecina del sector vecino, se ajusta al valor por defecto con el fin de inhibir interdependencias entre el sector actual y los sectores vecinos. Mientras que una actualización continua de probabilidades del símbolo del contexto puede tomar lugar justo como es el caso en el primer modo 20, la inicialización de las propiedades de símbolo del segundo modo 22 es independiente de cualquier sector previamente decodificado. Además, la decodificación predictiva se efectúa con restricción de la decodificación predictiva para no cruzar los límites de sector.
- Con el fin de facilitar el entendimiento de la descripción de la figura 24 y la siguiente descripción, se hace referencia a la figura 25 que muestra una implementación posible del decodificador 5 en un sentido más estructural que en comparación con la figura 24. Como es el caso en la figura 24, el decodificador 5 es un decodificador predictivo que utiliza decodificación de entropía adaptable al contexto para decodificación del flujo de datos, para obtener por ejemplo los parámetros de predicción y el residuo de predicción.
- Como se muestra en la figura 25, el decodificador 5 puede comprender un decodificador de entropía 24, un módulo de descuantificación y transformada inversa 26, un combinador 28 implementado como se muestra en la figura 25, por ejemplo, como un sumador y un predictor 28. El decodificador de entropía 24, módulo 26 y sumador 27 se conectan en serie entre la entrada y salida del decodificador 5 en el orden de su mención y el predictor 28 se conecta entre una salida del sumador 28 y una entrada adicional del mismo con el fin de formar el bucle de predicción junto con el combinador 27. Por lo tanto, el decodificador 24 tiene su salida conectada adicionalmente a una entrada del parámetro de codificación del predictor 28.
- Aunque la figura 25 da la impresión de que el decodificador decodifica en serie la instantánea actual, el decodificador 5 puede por ejemplo implementarse para decodificar la instantánea 10 en paralelo. El decodificador 5 puede comprender por ejemplo múltiples núcleos funcionando cada uno según los elementos 24-28 de la figura 25. El procesamiento en paralelo es sin embargo opcional y un decodificador que funciona en serie 5 es también apto para decodificar el flujo de datos interno en la entrada del decodificador de entropía 24.
- Con el fin de obtener eficientemente la habilidad recién mencionada ya sea de decodificación en serie o en paralelo de la instantánea actual 10, el decodificador 5 funciona en unidades de bloques de decodificación 30 con el fin de decodificar la instantánea 10. Los bloques de codificación 30 son por ejemplo bloques de hojas en los cuales se dividen bloques de árbol de codificación o bloques de codificación más grandes 32 mediante división de múltiples árboles recursiva tal como división de árbol cuádruple. Los bloques de árbol de codificación 32 a su vez, pueden disponerse de manera regular en columnas y filas para formar una división regular de la instantánea 10 en estos bloques de árbol de codificación 32. En la figura 25, los bloques de árbol de codificación 32 se muestran con líneas continuas, mientras que los bloques de codificación 30 se muestran con líneas discontinuas. Con fines de ilustración solamente un bloque de árbol de codificación 32 se muestra para dividirse adicionalmente en bloques de codificación 30, mientras que los otros bloques de árbol de codificación 32 se muestran para no dividirse adicionalmente para formar directamente un bloque de codificación en su lugar. El flujo de datos 12 puede comprender una porción de sintaxis que señala cómo una instantánea 10 se divide en los bloques de codificación 30.
- El flujo de datos 12 transporta, para cada bloque de codificación 30, elementos de sintaxis que revelan cómo los módulos 24 y 28 van a recuperar el contenido de instantánea dentro de ese bloque de codificación 30. Por ejemplo, estos elementos de sintaxis comprenden:
- 1) opcionalmente, dividir datos que dividen adicionalmente el bloque de codificación 30 en bloques de predicción,
 - 2) opcionalmente, dividir datos que dividen adicionalmente el bloque de codificación 30 en bloques residuales y/o de transformada,
 - 3) un modo de predicción que señala qué modo de predicción va a usarse para derivar la señal de predicción para el bloque de codificación 30, en el que la granularidad en la cual se señala este modo de predicción puede depender de los bloques de codificación 30 y/o del bloque de predicción.
 - 4) los parámetros de predicción pueden señalizarse por bloque de codificación o si está presente, por bloques de predicción con una clase de parámetros de predicción enviados, dependiendo, por ejemplo, del modo de predicción. Los modos de predicción posibles pueden comprender por ejemplo intra-predicción y/o inter-predicción.
 - 5) Otros elementos de sintaxis pueden también estar presentes tales como información de filtración para filtrar la instantánea 10 en el bloque de codificación 30 para obtener la señal de predicción y/o la señal reconstruida a reproducirse.
 - 6) Finalmente, información residual en forma de, entre otros, coeficientes de transformada puede estar

comprendida en un flujo de datos para el bloque de codificación 30; en unidades de bloques residuales, datos residuales pueden señalizarse, por bloque residual, la descomposición espectral puede por ejemplo realizarse en unidades de bloques de transformada mencionados anteriormente, si están presentes.

- 5 El decodificador de entropía 24 es responsable de obtener los elementos de sintaxis mencionados anteriormente del flujo de datos. Para este fin, el decodificador de entropía 24 usa decodificación de entropía adaptable al contexto. Esto es, el decodificador de entropía 24 proporciona varios contextos. Con el fin de derivar un cierto elemento de sintaxis del flujo de datos 12, el decodificador de entropía 24 selecciona un cierto contexto de entre los contextos posibles. La selección entre los posibles contextos se efectúa dependiendo del atributo de una vecindad de una porción de instantánea 10 a la cual el elemento de sintaxis actual pertenece. Para cada uno de los contextos posibles, el decodificador de entropía 24 gestiona probabilidades de símbolos, esto es, una estimación de la probabilidad para cada símbolo posible del alfabeto de símbolos en el cual se basa el funcionamiento del decodificador 24. La "gestión" implica las actualizaciones continuas anteriormente mencionadas de las probabilidades de símbolo de los contextos para adaptar las probabilidades de símbolos asociadas con cada contexto al contenido de instantánea real. Mediante esta medida, las probabilidades de símbolo se adaptan a las estadísticas de probabilidad reales de los símbolos.

- Otra circunstancia en donde los atributos de una vecindad influyen en la reconstrucción de una porción actual de la instantánea 10 tales como un bloque de codificación actual 30, es la decodificación predictiva dentro del predictor 28. La predicción está restringida no solamente a contenido de predicción dentro del bloque de codificación actual 30, sino que puede también abarcar predicción de parámetros contenidos dentro del flujo de datos 12 para el bloque de codificación actual 30, tales como parámetros de predicción, datos de división o incluso coeficientes de transformada. Esto es, el predictor 28 puede predecir el contenido de instantánea o tales parámetros de la vecindad mencionada anteriormente para obtener la señal escrita que se combina a continuación con el residuo de predicción como se obtiene por el módulo 26 del flujo de datos 12. En el caso de parámetros de predicción, el predictor 28 puede usar elementos de sintaxis contenidos dentro del flujo de datos como residuos de predicción para obtener el valor real del parámetro de predicción. El predictor 28 usa el último valor del parámetro de predicción con el fin de obtener la señal de predicción mencionada anteriormente que va a combinarse con el residuo de predicción en el combinador 27.

- La "vecindad" mencionada anteriormente cubre principalmente la parte superior izquierda de la circunferencia de la porción actual a la cual pertenece el elemento de sintaxis que va a decodificarse actualmente por entropía o el elemento de sintaxis que va a predecirse actualmente. En la figura 25, tal vecindad se ilustra en 34 a modo de ejemplo para un bloque de codificación 30.

- Un orden de codificación/decodificación se define entre los bloques de codificación 30: a un nivel más burdo, los bloques de árbol de codificación 32 de la instantánea 10 se exploran en un orden de exploración 36, en este caso ilustrado como una exploración de trama que va de fila a fila de la parte superior a la parte inferior. Dentro de cada bloque de árbol de codificación, los bloques de codificación 30 se exploran en un primer orden transversal de profundidad de tal manera que, cada nivel de jerarquía, el bloque de árbol de codificación 32 se explora sustancialmente también en una exploración de trama que va de fila a fila de la parte superior a la parte inferior.

- El orden de codificación definido entre los bloques de codificación 30 armoniza con la definición de la vecindad 34 usada para derivar un atributo en la vecindad para seleccionar contextos y/o efectuar predicción espacial en que la vecindad 34 cubre en su mayoría porciones de la instantánea 10 que ya han sido sometidas a decodificación según el orden de codificación. Siempre que una porción de la vecindad 34 cubre porciones no disponibles de la instantánea 10, se usan en lugar de esto datos por defecto, por ejemplo. Por ejemplo, una plantilla de vecindad 34 se puede extender fuera de la instantánea 10. Otra posibilidad es, sin embargo, que la vecindad 34 se extienda a un sector vecino.

- Los sectores dividen, por ejemplo, la instantánea 10 a lo largo de un orden de codificación/decodificación definido a lo largo de los bloques de codificación 30, esto es, cada sector es una secuencia no interrumpida continua de bloques de codificación 30 a lo largo del orden del bloque de codificación mencionado anteriormente. En la figura 25, los sectores se indican con líneas de puntos discontinuos 14. El orden definido entre los sectores 14 resulta de su composición de ejecuciones de bloques de codificación secuenciales 30 como se ha resumido anteriormente. Si la porción de elemento de sintaxis 18 de un cierto sector 14 indica que el mismo se va a decodificar en el primer modo, el decodificador de entropía 24 permite que la decodificación de entropía adaptable al contexto derive contextos a través de límites de sector. Esto es, la vecindad espacial 34 se usa con el fin de seleccionar contextos en datos de decodificación de entropía concernientes con el sector actual 14. En el caso de la figura 25, por ejemplo, el número de sector 3 puede ser el sector decodificado actualmente y en el elemento de sintaxis de decodificación de entropía concerniente con el bloque de codificación 30 o alguna porción contenida en el mismo, el decodificador de entropía 24 puede usar atributos que se derivan de porciones de decodificación dentro del sector vecino tal como el número de sector 1. El predictor 28 se comporta de la misma manera: para sectores que son del primer modo 20, el predictor 28 usa predicción espacial a través del límite de sector que circunda el sector actual.

Para sectores, sin embargo, que tienen el segundo modo 22 asociado con los mismos, esto es, para los cuales la

- porción de elemento de sintaxis 18 indica el segundo modo 22, el decodificador de entropía 24 y predictor 28 restringen la derivación de contextos de entropía y decodificación predictiva para depender de atributos concernientes con porciones que solo se encuentran dentro del sector actual. Obviamente, la eficacia de codificación manifiesta esta restricción. Por otra parte, los sectores del segundo modo 22 permiten interrumpir interdependencias entre las
- 5 secuencias de sector. Por lo tanto, los sectores del segundo modo 22 pueden estar intercalados dentro de la instantánea 10 o dentro de un video al cual la instantánea 10 pertenece con el fin de permitir la resincronización de puntos. No es necesario, sin embargo, que cada instantánea 10 tenga al menos un sector en el segundo modo 22.
- Como ya se mencionó anteriormente, los modos 20 y 22 primero y segundo también difieren en su inicialización de las probabilidades de símbolo. Los sectores codificados en el segundo modo 22, dan como resultado que el
- 10 decodificador de entropía 24 reinicialice las probabilidades independientemente de cualquier sector previamente decodificado, esto es, previamente decodificado en el sentido del orden definido entre los sectores. Las probabilidades de símbolo se ajustan, por ejemplo, a valores predeterminados conocidos tanto para el lado del codificador como el
- 15 lado del decodificador, o valores de inicialización que están contenidos dentro de sectores codificados en el segundo modo 22.
- Esto es, para sectores que se codifican/decodifican en un segundo modo 22, la adaptación de las probabilidades de símbolo siempre se inicia inmediatamente del comienzo de estos sectores. Por consiguiente, la exactitud de adaptación es mala para estos sectores al comienzo de estos sectores.
- 20 Las cosas son diferentes en sectores codificados/decodificados en el primer modo 20. Para los últimos sectores, la inicialización de las probabilidades de símbolo efectuadas por el decodificador de entropía 24 depende de los estados guardados de probabilidades de símbolo de un sector previamente decodificado. Siempre que un sector codificado/decodificado en un primer modo 20, tiene su comienzo, por ejemplo colocado en un lugar diferente del lado
- 25 izquierdo de la instantánea 10, esto es, no el lado desde donde la exploración de trama 36 comienza a recorrer de fila en fila antes de escalar a la siguiente fila hacia el fondo, se adaptan las probabilidades de símbolo como resultan al final de la decodificación de entropía del sector inmediatamente precedente. Esto se ilustra, por ejemplo, en la figura 2 por una flecha 38 para el sector número 4. El sector número 4 tiene su comienzo en algún lugar entre el lado derecho y lado izquierdo de la instantánea 10 y por lo tanto, en la inicialización de las probabilidades de símbolo, el
- 30 decodificador de entropía 24 adopta, en la inicialización de las probabilidades de símbolo, las probabilidades de símbolo que se obtuvieron en la decodificación de entropía del sector inmediatamente precedente, esto es sector número 3, hasta el final del mismo, esto es, incluyendo la actualización continua de probabilidades de símbolo durante la decodificación de entropía del sector 3 hasta su final.
- Los sectores que tienen el segundo modo 22 asociado con los mismos, que tienen, sin embargo, el comienzo en el
- 35 lado izquierdo de la instantánea 10, tal como por ejemplo, el sector número 5 no adaptan las probabilidades de símbolo tal como se obtienen después de terminar la decodificación de entropía del sector inmediatamente precedente número 4, debido a que esto impediría que el decodificador 5 descodifique en paralelo la instantánea 10 mediante el uso de procesamiento de frente de onda. Más bien, como se ha resumido anteriormente, el decodificador de entropía 24
- 40 adapta las probabilidades de símbolo como se obtienen después de terminar la decodificación de entropía del segundo orden en, codificación/decodificación 36, bloque de árbol de codificación 32 en la fila del bloque de árbol de codificación, en el orden de codificación/decodificación 36 inmediatamente precedente, como se ilustra por la flecha 40.
- En la figura 25, por ejemplo, la instantánea 10 se dividió a modo de ejemplo en tres filas de bloques de árbol de
- 45 codificación de cuatro columnas de bloques de raíz de árbol de codificación 32 y cada fila de bloque de árbol de codificación se subdividió en dos sectores 14, de tal manera que el comienzo de cada segundo sector coincide con la primera unidad de codificación en orden unitario de codificación de una fila de bloque de raíz de árbol de codificación respectivo. El decodificador de entropía 24, por consiguiente, sería apto para usar el procesamiento de frente de onda
- 50 en la decodificación de la instantánea 10, mediante la decodificación de cada fila de bloque de raíz de árbol de codificación en paralelo, con el comienzo de la decodificación de estas filas de bloque de raíz árbol de codificación de manera escalonada, empezando con una primera fila de bloque de raíz árbol de codificación o más superior, luego la segunda y luego la tercera.
- Naturalmente, la división de los bloques 32 de manera recursiva a bloques de codificación adicionales 30 es opcional
- 55 y por lo tanto, en un sentido más general, los bloques 32 podrían ser llamados "bloques de codificación" también. Esto es, hablando más en general, la instantánea 10 puede dividirse en bloques de codificación 32 dispuestos en filas y columnas y que tienen un orden de exploración de trama 36 definido entre sí y el decodificador 5 puede considerarse asociado a cada sector 14 con un subconjunto continuo de los bloques de codificación 32 en el orden de exploración
- 60 de trama 36, de tal manera que los subconjuntos se siguen entre sí a lo largo del orden de exploración de trama 36 según el orden de sector.

Como se aclaró del análisis anterior, el decodificador 5 o más específicamente, el decodificador de entropía 24 puede estar configurado para guardar probabilidades de símbolos tal como se obtienen en la decodificación de entropía

- adaptable al contexto de cualquier sector hasta un segundo bloque de codificación en una fila de bloques de codificación según el orden de exploración de trama 36. En la inicialización de las propiedades del símbolo para la decodificación de entropía adaptable al contexto de un sector actual que tiene el primer modo 20 asociado con el mismo, el decodificador 5 o más específicamente, el decodificador de entropía 24, verifica en cuanto a si un primer
- 5 bloque de codificación 32 del subconjunto continuo de bloques de codificación 32 asociados con el sector actual es un primer bloque de codificación 32 en una fila de bloques de codificación según el orden de exploración de trama 36. Si es así, las probabilidades de símbolo para la decodificación de entropía adaptable al contexto del sector actual se inicializan como se explica con respecto a la flecha 40, es decir dependiendo de las probabilidades de símbolo guardadas como se obtienen en la decodificación de entropía de contexto del sector previamente decodificado hasta
- 10 el segundo bloque de codificación en la fila de bloques de codificación según el orden de exploración de trama 36. Si no, la inicialización de las probabilidades de símbolo para la decodificación de entropía adaptable al contexto del sector actual se efectúa dependiendo de las probabilidades de símbolo como se obtienen en la decodificación de entropía adaptable al contexto del sector decodificado previamente hasta el final del sector previamente decodificado, esto es, según la flecha 38. Otra vez, en el caso de inicialización según 38, se quiere dar a entender el estado guardado al
- 15 final de la decodificación de entropía del sector inmediatamente precedente en el orden de sector 36, mientras que en el caso de la inicialización 40, es el sector previamente decodificado que comprende el final del segundo bloque de la fila inmediatamente precedente del bloque 32 en el orden de bloque 36.
- Como se ilustra por líneas discontinuas en la figura 24, el decodificador puede estar configurado para responder a la
- 20 porción de elemento de sintaxis 18 dentro del sector actual de los sectores 14 para decodificar el sector actual según uno de al menos tres modos. Esto es, puede haber un tercer modo 42 además de los otros 20 y 22. El tercer modo 42 puede diferir del segundo modo 22 en que se permite la predicción a través de los límites de sector, mientras que la codificación/decodificación de entropía todavía se restringe para no cruzar límites de sector.
- 25 Anteriormente, se presentaron dos realizaciones con respecto a la porción de elemento de sintaxis 18. La tabla a continuación resume estas dos realizaciones.

	realización 1	realización 2
porción de elemento de sintaxis	dependent_slice_flag, no_cabac_reset_flag	dependent_slice_flag
modo 1	dependent_slice_flag = 1, no_cabac_reset_flag = 1	dependent_slice_flag = 1,
modo 2	dependent-slice-flag = 0	dependent_slice_flag = 0
modo 3	dependent_slice_flag = 1, no_cabac_reset_flag = 0	dependent_slice_flag = 1, cabac_independent_flag = 1, tiles_or_entropy_coding_sync_idc = 3

- En una realización, la porción de elemento de sintaxis 18 se forma por dependent_slice_flag individualmente, mientras
- 30 que, en la otra realización, la combinación de dependent_slice_flag y no_cabac_reset_flag forma la porción del elemento de sintaxis. Se hace referencia al proceso de sincronización para variables de contexto en cuanto a lo que se refiere a la inicialización de probabilidades de símbolo dependiendo de estados guardados de probabilidades de símbolo de un sector previamente decodificado. En particular, el decodificador puede estar configurado para, si last_ctb_cabac_inic_flag=0 y tiles_or_entropy_coding_sync_idc=2, las probabilidades de símbolo guardados como se
- 35 obtienen en la decodificación de entropía adaptable al contexto del sector previamente decodificado hasta un segundo bloque de codificación en una fila según el orden de exploración de trama y en la inicialización de las probabilidades de símbolo para la decodificación de entropía adaptable al contexto del sector actual según el primer modo, verifica si un primer bloque de codificación del subconjunto continuo de bloques de codificación asociados con el sector actual es un primer bloque de codificación en una fila según el orden de exploración de trama y si es así, inicializar las
- 40 propiedades de símbolo para la decodificación adaptable al contexto del sector actual dependiendo de las propiedades de símbolo guardadas como se obtienen en la decodificación de entropía adaptable al contexto del sector previamente decodificado a un segundo bloque de codificación en una fila según el orden de exploración de trama y si no, inicializar las probabilidades de símbolo para la decodificación de entropía adaptable al contexto del sector actual dependiendo de las propiedades de símbolo como se obtienen en la decodificación de entropía adaptable al contexto del sector
- 45 previamente decodificado al final del sector previamente decodificado.

- Por lo tanto, en otras palabras, según la segunda realización para una sintaxis, el decodificador reconstruiría la instantánea 10 a partir del flujo de datos 12 al que la instantánea se codifica en unidades de sectores 14 en las cuales se divide la instantánea (10), en la que el decodificador está configurado para decodificar los sectores 14 del flujo de
- 50 datos 12 según el orden de sector 16 y el decodificador responsable de una porción de elemento de sintaxis 18, es decir, dependent_slice_flag dentro de un sector actual de los sectores, para decodificar el sector actual según uno de al menos dos modos 20, 22. Según uno primero 20 de los al menos dos modos, en concreto dependent_slice_flag=1,

el decodificador decodifica el sector actual del flujo de datos 12 utilizando decodificación de entropía adaptable al contexto 24 que incluye una derivación de contextos a través de límites de sector, una actualización continua de probabilidades de símbolo de los contextos y una inicialización 38, 40 de las probabilidades de símbolo dependiendo de los estados guardados de probabilidades de símbolo de un sector previamente decodificado y decodificación predictiva a través de los límites de sector y según uno segundo 22 de al menos dos modos, es decir si `dependent_slice_flag=0`, el decodificador decodifica el sector actual del flujo de datos 12 utilizando decodificación de entropía adaptable al contexto con restricción de la derivación de los contextos para no cruzar los límites de sector, una actualización continua de probabilidades de símbolo de los contextos y una inicialización de las probabilidades de símbolo independientemente de cualquier sector previamente decodificado y decodificación predictiva con restricción de la decodificación predictiva para no cruzar los límites de sector. La instantánea 10 puede dividirse en bloques de codificación 32 dispuestos en filas y columnas y que tiene un orden de exploración de trama 36 definidos entre sí y el decodificador está configurado para asociar cada sector 14 con un subconjunto continuo de los bloques de codificación 32 en el orden de exploración de trama 36, de tal manera que los subconjuntos se siguen entre sí a lo largo del orden de exploración de trama 36 según el orden de sector. El decodificador puede estar configurado para, esto es, en respuesta a `tiles_or_entropy_coding_sync_idc=2`, guardar probabilidades de símbolo como se obtienen en la decodificación de entropía adaptable al contexto del sector decodificado previamente hasta un segundo bloque de codificación 32 en una fila según el orden de exploración de trama 36 y en la inicialización de las probabilidades de símbolo para la decodificación de entropía adaptable al contexto del sector actual según el primer modo, verificar si un primer bloque de codificación del subconjunto continuo de bloques de codificación 32 asociados con el sector actual es un primer bloque de codificación 32 en una fila según el orden de exploración de trama y si es así, inicializar 40 las probabilidades de símbolo para la decodificación de entropía adaptable al contexto del sector actual dependiendo de las probabilidades de símbolo guardadas como se obtienen en la decodificación de entropía adaptable al contexto del sector previamente decodificado a un segundo bloque de codificación en una fila según el orden de exploración de trama 36 y si no, inicializar 38 las probabilidades de símbolo para la decodificación de entropía adaptable al contexto del sector actual dependiendo de las probabilidades de símbolo como se obtienen en la decodificación de entropía adaptable al contexto del sector previamente decodificado hasta el final del sector previamente decodificado. El decodificador puede estar configurado para responder a la porción del elemento de sintaxis (18) dentro del sector actual de los sectores 14 para decodificar el sector actual según al menos uno de tres modos, es decir, en el uno del primer 20 y un tercer modo 42 o un segundo modo 22, en el que el decodificador está configurado para, según el tercer modo 42, es decir si **`dependent_slice_flag=1`** y `tiles_or_entropy_coding_sync_idc=3`, decodificar el sector actual del flujo de datos utilizando decodificación de entropía adaptable al contexto con restricción de la derivación de los contextos para no cruzar los límites de sector, una actualización continua de probabilidades de símbolo de los contextos y una inicialización de las probabilidades de símbolo independientemente de cualquier sector previamente decodificado y decodificación predictiva a través de los límites de sector, en el que el uno del primer y tercer modos se selecciona dependiendo de un elemento de sintaxis, es decir, `cabac_independent_flag`. El decodificador puede estar configurado para, concretamente, si `tiles_or_entropy_coding_sync_idc = 0, 1 y 3` ("3" cuando `cabac_independent_flag = 0`), guardar probabilidades de símbolo tal como se obtienen en decodificación de entropía adaptable al contexto del sector previamente decodificado hasta el final del sector previamente decodificado y en la inicialización de las probabilidades de símbolo para la decodificación de entropía adaptable al contexto del sector actual según el primer modo, inicializar las probabilidades de símbolo para la decodificación de entropía adaptable al contexto del sector actual dependiendo de las probabilidades de símbolo guardadas. El decodificador puede estar configurado para, concretamente, si `tiles_or_entropy_coding_sync_idc=1`, en el primer y segundo modo, restringir la decodificación predictiva en mosaicos en los cuales se subdivide la instantánea.

45 Naturalmente, un codificador puede ajustar la sintaxis presentada anteriormente con el fin de permitir por consiguiente que el decodificador obtenga las ventajas resumidas anteriormente. El codificador puede ser un codificador de procesamiento en paralelo, tal como un codificador de múltiples núcleos, pero no es necesario. Para la codificación de la instantánea 10 en el flujo de datos 12 en unidades de sector 12, el codificador estaría configurado para codificar los sectores 14 en el flujo de datos 12 según el orden de sector 16. El codificador determinaría la porción del elemento de sintaxis 18 para codificar el mismo a un sector actual de los sectores, de tal manera que la porción del elemento de sintaxis señala el sector actual que va a codificarse según uno de los al menos dos modos 20, 22 y si el sector actual va a codificarse según un primer modo 20 de los al menos dos modos, codificar el sector actual en el flujo de datos 12 usando codificación de entropía adaptable al contexto 24 que incluye una derivación de contextos a través de límites de sector, una actualización continua de probabilidades de símbolo de los contextos y una inicialización 38, 40 de las probabilidades de símbolo dependiendo de los estados guardados de probabilidades de símbolo de un sector previamente codificado y codificación predictiva a través de los límites de sector y si el sector actual va a codificarse según el segundo 22 de los al menos dos modos, codificar el sector actual en el flujo de datos 12 usando codificación de entropía adaptable al contexto con restricción de la derivación de los contextos para no cruzar los límites de sector, una actualización continua de probabilidades de símbolo de los contextos y una inicialización de las probabilidades de símbolo independientemente de cualquier sector previamente codificado y codificación predictiva con restricción de la codificación predictiva para no cruzar los límites de sector. Aunque la instantánea 10 puede dividirse en bloques de codificación 32 dispuestos en filas y columnas y que tienen un orden de exploración de trama 36 definido entre ellas, el codificador puede estar configurado para asociar cada sector 14 con un subconjunto continuo de los bloques de codificación 32 en el orden de exploración de trama 36, de tal manera que los subconjuntos se siguen entre sí a lo

largo del orden de exploración de trama 36 según el orden de sector. El codificador puede estar configurado para guardar probabilidades de símbolo tal como se obtienen en codificación de entropía adaptable al contexto del sector previamente codificado hasta un segundo bloque de codificación 32 en una fila, según el orden de exploración de trama 36 y en la inicialización de las probabilidades de símbolo para la codificación de entropía adaptable al contexto del sector actual según el primer modo, verificar en cuanto a si un primer bloque de codificación del subconjunto continuo de bloques de codificación 32 asociados con el sector actual es un primer bloque de codificación 32 en una fila según el orden de exploración de trama y si es así, inicializar 40 las probabilidades de símbolo para la codificación de entropía adaptable al contexto del sector actual, dependiendo de las probabilidades de símbolo guardadas, tal como se obtienen en codificación de entropía adaptable al contexto del sector previamente codificado, hasta un segundo bloque de codificación en una fila, según el orden de exploración de trama 36 y si no, inicializar 38 las probabilidades de símbolo para la codificación de entropía adaptable al contexto del sector actual, dependiendo de las probabilidades de símbolo, tal como se obtienen en codificación de entropía adaptable al contexto del sector previamente decodificado hasta el final del sector previamente codificado. El codificador puede estar configurado para codificar la porción de elemento de sintaxis (18) en el sector actual de los sectores (14) de tal manera que el sector actual se señala para codificarse en el mismo según uno de los al menos dos modos, es decir en uno del primero (20) y un tercer modo (42) o un segundo modo (22), en el que el codificador está configurado para ** según el tercer modo (42), codificar el sector actual en el flujo de datos usando codificación de entropía adaptable al contexto con restricción de la derivación de los contextos para no cruzar los límites de sector, una actualización continua de probabilidades de símbolo de los contextos y una inicialización de las probabilidades de símbolo independientemente de cualquier sector previamente codificado y codificación predictiva a través de los límites de sector, en el que el codificador distingue entre uno del primer y tercer modos al ser un elemento de sintaxis, es decir `cabac_independent_flag`, por ejemplo. El codificador puede estar configurado para determinar un elemento de sintaxis genérico, tal como `dependent_slices_present_flag` y escribir el mismo al flujo de datos haciendo funcionar en uno de los al menos dos modos de operación genéricos, dependiendo del elemento de sintaxis genérico, concretamente, según un primer modo de operación genérico, efectuar la codificación de la porción de elemento de sintaxis para cada sector y según un segundo modo de operación genérico, inevitablemente usando un modo diferente de los al menos dos modos diferentes del primer modo. El codificador puede estar configurado según el primero y segundo modos, inevitablemente y continuar de manera ininterrumpida la actualización continua de las probabilidades de símbolo desde el comienzo hasta el final del sector actual. El codificador puede estar configurado para guardar probabilidades de símbolo tal como se obtienen en codificación de entropía adaptable al contexto del sector previamente codificado hasta un final del sector previamente codificado y en la inicialización de las probabilidades de símbolo para la codificación de entropía adaptable al contexto del sector actual según el primer modo, inicializar las probabilidades de símbolo para la codificación de entropía adaptable al contexto del sector actual, dependiendo de las probabilidades de símbolo guardadas. El codificador puede además, en los modos primero y segundo, restringir la codificación predictiva en mosaicos en los que se subdivide la instantánea.

Una estructura posible de un codificador se ilustra en la figura 26 por motivos de completitud. El predictor 70 funciona casi de la misma manera que el predictor 28, esto es, efectúa la predicción, pero también determina, mediante optimización, por ejemplo, los parámetros de codificación que incluyen los parámetros y modos de predicción. Los módulos 26 y 27 también se producen en el decodificador. El restador 72 determina el residuo de predicción sin pérdidas que a continuación, se codifica con pérdidas, mediante el uso de cuantificación y opcionalmente, usando transformación de descomposición espectral, en el módulo de transformación y cuantificación 74. El codificador de entropía 76 efectúa la codificación de entropía adaptable al contexto.

Además de los ejemplos concretos de sintaxis anteriores, se resume un ejemplo diferente a continuación que presenta la concordancia entre los términos usados posteriormente y términos usados anteriormente.

En particular, sin haber sido señalados en particular anteriormente en el presente documento, los sectores dependientes no son solamente "dependientes" porque los mismos permitan aprovechar el conocimiento conocido desde el exterior de su límite, tal como se ha resumido anteriormente, sino que tienen los contextos de entropía adaptados más rápido u obtienen una predicción espacial mejor debido a permitir el cruce de su límite. En lugar de esto, con el fin de ahorrar coste de tasa que se tiene que gastar para definir los encabezamientos de sector al dividir la instantánea en sectores, los sectores dependientes adoptan una parte de la sintaxis de encabezamiento de sector de los sectores previos, esto es, esta porción de encabezamiento de sintaxis de sector no se transmite otra vez para sectores dependientes. Esto se muestra por ejemplo en la figura 16 con el número 100 y en la figura 21 con el número 102, según las cuales, el tipo de sector se adopta del sector previo, por ejemplo. Mediante esta medida, la subdivisión de una instantánea en sectores, tal como un sector independiente y sectores dependientes, es menos costosa en términos de gasto de consumo de bits.

Es la recién mencionada dependencia la que conduce, en el ejemplo que se señala posteriormente en el presente documento, a una terminología ligeramente diferente: los sectores se definen como porciones unitarias de una instantánea en la cual la sintaxis de encabezamiento de sector es ajustable individualmente. Por consiguiente, los sectores están compuestos por uno sector, usando la nomenclatura anterior, independiente/regular/normal, ahora llamado segmento de sector independiente y no uno o más sectores, usando la nomenclatura anterior, dependientes,

ahora llamados segmentos de sectores dependientes.

La figura 27 muestra, por ejemplo, una instantánea que va a dividirse en dos sectores, uno formado por segmentos de sector 14₁ a 14₃ y la otra formada solamente por el segmento de sector 14₄. Los índices 1 a 4 muestran el orden de sector en orden de codificación. Las figuras 28a y b muestran un ejemplo diferente en caso de una subdivisión de una instantánea 10 en dos mosaicos, que presentan, en el caso de la figura 28a, un sector formado por todos los cinco segmentos de sector 14, que cubren ambos mosaicos 50₁ y 50₂, el índice surge otra vez en el orden de codificación, y, en el caso de la figura 28a, dos sectores formados por los segmentos de sector 14₁ y 14₂ y 14₃ y 14₄ respectivamente, que subdividen el mosaico 50₁ y otro sector formado por los segmentos de sector 14₅ - 14₆ que cubren el mosaico 50₂.

Las definiciones podrían ser como sigue:

segmento de sector dependiente: un *segmento de sector* para el cual los valores de algunos *elementos de sintaxis* del *encabezamiento de segmento de sector* se infieren de los valores para el *segmento de sector independiente* precedente en orden de decodificación, anteriormente, en las realizaciones anteriores, llamado un *sector dependiente*.

segmento de sector independiente: un *segmento de sector* para el cual los valores de los elementos de sintaxis del *encabezamiento de segmento de sector* no se infieren de los valores para un segmento de sector precedente, anteriormente, en las realizaciones anteriores, llamado un *sector normal*.

sector: un número entero de *unidades de árbol de codificación* contenidas en un *segmento de sector independiente* y todos los *segmentos de sectores dependientes* siguientes (si los hay) que preceden al siguiente *segmento de sector independiente* (si lo hay) dentro de la misma *unidad de acceso/instantánea*.

encabezamiento de sector: el *encabezamiento de segmento de sector* del *segmento de sector independiente* que es un *segmento de sector* actual o es el *segmento de sector independiente* que precede a un *segmento de sector dependiente* actual.

segmento de sector: un número entero de *unidades de árbol de codificación* ordenadas consecutivamente en la *exploración de mosaicos* y contenidas en una sola *unidad de NAL*; la división de cada *instantánea* en segmentos de sector es una *división*.

encabezamiento de segmento de sector: una parte de un *segmento de sector* codificado que contiene los elementos de datos pertenecientes a la primera o todas las *unidades de árbol de codificación* representadas en el *segmento de sector*.

La señalización de los “modos” 20 y 22, esto es “segmento de sector dependiente” y “segmento de sector independiente” podría ser como sigue:

En algunas unidades de NAL extra, tales como PPS, un elemento de sintaxis podría usarse para señalar si se usan sectores dependientes o no para una cierta instantánea de incluso una secuencia para ciertas instantáneas:

dependent_slice_segments_enabled_flag igual a 1 especifica la presencia del elemento de sintaxis *dependent_slice_segment_flag* en los encabezamientos de segmento de sector. *dependent_slice_segments_enabled_flag* igual a 0 especifica la ausencia del elemento de sintaxis *dependent_slice_segment_flag* en los encabezamientos de segmento de sector.

dependent_slice_segments_enabled_flag es similar en alcance a **dependent_slices_present_flag** descrito anteriormente.

De manera similar, **dependent_slice_flag** podría llamarse **dependent_slice_segment_flag** para tener en cuenta la diferente nomenclatura con respecto a sectores.

dependent_slice_segment_flag igual a 1 especifica que el valor de cada elemento de sintaxis de encabezamiento de segmento de sector que no está presente en el encabezamiento del segmento de sector actual se infiere para que sea igual al valor del elemento de sintaxis de encabezamiento de segmento de sector correspondiente en el encabezamiento de sector, esto es, el encabezamiento de segmento de sector del segmento de sector independiente precedente.

En el mismo nivel, tal como el nivel de instantánea, el siguiente elemento de sintaxis podría incluirse:

entropy_coding_sync_enabled_flag igual a 1 especifica que un proceso de sincronización específico para

variables de contexto se invoca antes de la decodificación de la unidad de árbol de codificación que incluye el primer bloque de árbol de codificación de una fila de bloques de árbol de codificación en cada mosaico en cada instantánea que hace referencia a PPS y un proceso de almacenamiento específico para variables de contexto se invoca después de la decodificación de la unidad de árbol de codificación que incluye el segundo bloque de árbol de codificación de una fila de bloques de árbol de codificación en cada mosaico en cada instantánea que hace referencia al PPS. `entropy_coding_sync_enabled_flag` igual a 0 especifica que no se requiere ningún proceso de sincronización específico para variables de contexto a invocarse antes de la decodificación de la unidad de árbol de codificación que incluye el primer bloque de árbol de codificación de una fila de bloques de árbol de codificación en cada mosaico en cada instantánea que hace referencia al PPS y no se requiere ningún proceso de almacenamiento específico para variables de contexto a invocarse después de la decodificación de la unidad de árbol de codificación que incluye el segundo bloque de árbol de codificación de una fila de bloques de árbol de codificación en cada mosaico en cada instantánea que hace referencia al PPS.

Es un requisito de la conformación de flujo de bits que el valor de `entropy_coding_sync_enabled_flag` será el mismo para todos los PPS que están activados dentro de un CVS.

Cuando `entropy_coding_sync_enabled_flag` es igual a 1 y el primer bloque de árbol de codificación en un sector no es el primer bloque de árbol de codificación de una fila de bloques de árbol de codificación en un mosaico, es un requisito de conformación de flujo de bits que el último bloque de árbol de codificación en el sector pertenecerá a la misma fila de bloques de árbol de codificación como el primer bloque de árbol de codificación en el sector. Cuando `entropy_coding_sync_enabled_flag` es igual a 1 y el primer bloque de árbol de codificación en un segmento de sector no es el primer bloque de árbol de codificación de una fila de bloques de árbol de codificación en un mosaico, es un requisito de la conformación de flujo de bits que el último bloque de árbol de codificación en el segmento de sector pertenecerá a la misma fila de bloques de árbol de codificación que el primer bloque de árbol de codificación en el segmento de sector.

Como ya se describió, el orden de codificación/decodificación entre los CTB 30 conduce de fila en fila en manera de trama de arriba a abajo empezando con la exploración del primer mosaico, a continuación visitando el siguiente mosaico, si más de un mosaico está presente en la instantánea.

El decodificador 5, y, por consiguiente, el codificador, actúa como sigue en la decodificación (codificación) de entropía de los segmentos de sector 14 de la instantánea:

A1) Siempre que un elemento de sintaxis actualmente decodificado/codificado `synEl` es el primer elemento de sintaxis de un mosaico 50, el segmento de sector 14 o fila de CTB, se comienza un proceso de inicialización de la figura 29.

A2) De otra manera, la decodificación de este elemento de sintaxis tiene lugar utilizando los contextos de entropía actuales.

A3) Si el elemento de sintaxis actual fue el último elemento de sintaxis en un CTB 30, entonces se comienza un proceso de almacenamiento de contexto de entropía como se muestra en la figura 30.

A4) El proceso avanza a A1) con el siguiente elemento de sintaxis.

En el proceso de inicialización, se verifica 200 si `synEl` es el primer elemento de sintaxis de un segmento de sector 14 o mosaico 50. Si la respuesta es sí, los contextos se inicializan independientemente de cualquier segmento de sector previo en la etapa 202. Si la respuesta es no, se verifica 204 si `synEl` es el primer elemento de sintaxis de una fila de CTB 30 y `entropy_coding_sync_enabled_flag` es igual a uno. Si la respuesta es sí, se verifica 206 si en la línea previa de CTB 30 de mosaico igual, el segundo CTB 30 está disponible (véase la figura 23). Si la respuesta es sí, se efectúa una adopción de contexto según 40 en la etapa 210 utilizando las probabilidades de contexto almacenadas actualmente para adopciones de tipo 40. Si la respuesta es no, los contextos se inicializan independientemente de cualquier segmento de sector previo en la etapa 202. Si la verificación 204 revela que no, entonces se verifica en la etapa 212, si `synEl` es el primer elemento de sintaxis en el primer CTB de un segmento de sector dependiente 14 y si `dependent_slice_segment_flag` es igual a uno, y si la respuesta es sí, se efectúa la adopción de contexto según 38 en la etapa 214 utilizando las probabilidades de contexto almacenadas actualmente para adopciones de tipo 38. Después de cualquier etapa 214, 212, 210 y 202, la decodificación/codificación se comienza realmente.

Los segmentos de sectores dependientes con `dependent_slice_segment_flag` igual a uno, ayudan por lo tanto a disminuir adicionalmente el retardo de codificación/decodificación con casi ninguna penalización de la eficacia de codificación.

En el proceso de almacenamiento de la figura 30, se verifica en la etapa 300 si el `synEl` codificado/decodificado es el último elemento de sintaxis de un segundo CTB 30 de una fila de CTB 30, y `entropy_coding_sync_enabled_flag` es

igual a uno. Si la respuesta es sí, los contextos de entropía actuales se almacenan en la etapa 302, esto es, las probabilidades de codificación de entropía de los contextos, en un almacenamiento que es específico para adopciones en 40. De manera similar, se verifica en la etapa 304, además de las etapas 300 o 302, si el synEl codificado/decodificado es el último elemento de sintaxis de un segmento de sector 14, y
5 dependent_slice_segment_flag es igual a uno. Si la respuesta es sí, los contextos de entropía actuales se almacenan en la etapa 306, esto es, las probabilidades de codificación de entropía de los contextos, en un almacenamiento que es específico para adopciones en 38.

Se observará que cualquier interrogación de verificación en cuanto a si un elemento de sintaxis es el primer synEl de una fila de CTB, aprovecha por ejemplo, el elemento de sintaxis slice_address 400 dentro de los encabezamientos de segmento de sector, esto es, un elemento de sintaxis de inicio que revela una posición del comienzo del segmento de sector respectivo junto con el orden de decodificación.

En la reconstrucción de la instantánea 10 a partir del flujo de datos 12 utilizando procesamiento de WPP, el decodificador puede aprovechar exactamente la última porción de la sintaxis de inicio 400 para recuperar los puntos de entrada de subflujo WPP. Puesto que cada segmento de sector comprende una porción de sintaxis de inicio 400 que indica la posición de un comienzo de decodificación del segmento de sector respectivo dentro de la instantánea 10, el decodificador puede identificar los puntos de entrada de los subflujos de WPP en los que se agrupan los segmentos de sector, al identificar, utilizando las porciones de sintaxis de inicio de los segmentos de sector 400, segmentos de sector que inician en el lado izquierdo de la instantánea. El decodificador puede a continuación, en paralelo, decodificar los subflujos de WPP de manera escalonada con el comienzo secuencialmente de la decodificación de los subflujos de WPP según el orden de sector. Los segmentos de sector pueden aún ser más pequeños que la anchura de una instantánea, esto es una fila de CTB, de tal manera que su transmisión puede intercalarse entre los subflujos de WPP para reducir adicionalmente el retardo de extremo a extremo de transmisión global. El codificador proporciona cada sector (14) con una porción de sintaxis de inicio (400) que indica una posición del comienzo de codificación del sector respectivo dentro de la instantánea (10) y agruparía los sectores en subflujos de WPP de tal manera que para cada subflujo de WPP, el primer sector en orden de sectores se inicia en el lado izquierdo de la instantánea. El codificador puede aún, por sí mismo, usar el procesamiento de WPP en la codificación de la instantánea: el codificador codifica en paralelo los subflujos de WPP en de manera escalonada con el comienzo secuencialmente de la codificación de los subflujos de WPP según el orden de sector.

Mientras tanto, el último aspecto de usar las porciones de sintaxis de inicio del segmento de sector como medio para ubicar los puntos de entrada del subflujo de WPP, puede usarse sin el concepto de sector dependiente.

35 Sería factible para todo el procesamiento en paralelo de la instantánea 10, ajustar las variables anteriores como sigue:

	tiles_enabled_flag==0	tiles_enabled_flag==1	entropy_cooding_sync_enable_flag==1	tiles_enabled_flag==1	entropy_cooding_sync_enable_flag
dependent_slice_segment_flag	0/1	0/1	0/1	0/1	0/1
subdivisión de instantánea en mosaicos	No	Sí	No	Sí	No
Subdivisión de sector con puntos de entrada para indicar el inicio de subflujo de WPP o mosaico	No	Sí	Sí	No	No
procesamiento en paralelo posible	en sectores	en mosaicos	WPP	en mosaicos	WPP

40 Sería incluso factible mezclar WPP con la división de mosaicos. En ese caso, se podrían tratar mosaicos como instantáneas individuales: cada una de las cuales que usa WPP estaría compuesta de un sector que tiene uno o más segmentos de sectores dependientes y la verificación en la etapa 300 y 208 haría referencia al segundo CTB en la fila de CTB anterior en el mismo mosaico, tal como las etapas 204 y A1 harían referencia al primer CTB 30 en la fila de CTB 30 del mosaico actual. En ese caso, la tabla anterior podría extenderse:

	tiles_enabled_flag	tiles_enabled_flag==1	entropy_coding_sync_enable_flag==1	tiles_enabled_flag==1	entropy_coding_sync_enable_flag==1	entropy_coding_syncenable_flag==1	entropy_coding_sync_enable_flag==1
dependent_slice_segment_flag	0/1	0/1	0/1	0/1	0/1	0/1	0/1
Instantánea subdividida en mosaicos	No	Sí	No	Sí	No	Sí	Sí
Subdivisión de sector con puntos de entrada para indicar el inicio del subflujo de WPP o mosaico	No	Sí	Sí	No	No	No (cada subflujo de un mosaico en su propio sector dependiente)	Sí (cada mosaico en un sector separado y puntos de entrada para subflujos de WPP en el sector del mosaico)
procesamiento en paralelo posible	en sectores	en mosaicos	WPP	en mosaicos	WPP	subflujos de WPP en mosaicos	subflujos de WPP en mosaicos

Como una breve nota, la última extensión también habría sido posible con la realización 2. La realización 2 permite el siguiente procesamiento:

5

	nivel de sintaxis				
tiles_or_entropy_coding_sync_idc	por instantánea	0	1	2	3
cabac_independent_flag	Por sector	no presente	no presente	no presente	0/1
dependent_slice_flag	por sector	0/1	0/1	0/1	0/1
Subdivisión de instantánea en mosaicos		No	Sí	No	No
Subdivisión de mosaicos con puntos de entrada para indicar inicio de subflujo de WPP o mosaico		No	Sí	Sí	No
procesamiento en paralelo posible		en sectores	en mosaicos	WPP	Codificación de entropía de sectores

Sin embargo con las siguientes extensiones, la tabla daría como resultado:

Agregar a semántica de conjunto de parámetros de instantánea:

10

Si tiles_or_entropy_coding_sync_idc es igual a 4, cada una, excepto la primera fila de CTB estará contenida en un sector diferente con el indicador de sector dependiente ajustado a 1. Los CTB de diferentes filas no deben estar presentes en el mismo sector. Puede haber más de un sector presente por fila de CTB.

15

Si tiles_or_entropy_coding_sync_idc es igual a 5, los CTB de cada uno, excepto el primer mosaico debe estar contenidos en un sector diferente. Los CTB de diferentes mosaicos no deben estar presentes en el mismo sector. Puede haber más de un sector presente por mosaico.

Véase la figura 31, para explicación adicional.

20

Esto es, la tabla anterior puede extenderse:

	Nivel de sintaxis								
tiles_or_entropy_ coding_sync_idc	por instantánea	0	1	2	3	5	4	6	7
cabac_ independent_flag	por mosaico	no presente	no presente	no presente	0/1	no presente	no presente	no presente	no presente
dependent_slice flag	por mosaico	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1
Subdivisión de instantánea en mosaicos		No	Sí	No	No	Sí	No	Sí	Sí
Subdivisión de sector con puntos de entrada para indicar inicio de subflujo o mosaico de WPP.		No	Sí	Sí	No	No	No	No (cada subflujo de un mosaico en su propio sector dependiente)	Sí (cada mosaico en un sector separado y puntos de entrada para subflujo de WPP en el sector del mosaico)
procesamiento en paralelo posible		en sectores	en mosaicos	WPP	codificaci ón de entropía de sectores	en mosaic os	WPP	Subflujo de WPP en mosaicos	Subflujo de WPP en mosaicos

Con respecto a las realizaciones anteriores, debe observarse que el decodificador podría estar configurado por ejemplo para responder a tiles_or_entropy_coding_sync_idc=1,2, en los modos primero y segundo, leer información del sector actual que revela una subdivisión del sector actual en subsecciones paralelas, en el que las subsecciones paralelas podrían ser subflujos de WPP o mosaicos, detener la decodificación de entropía adaptable al contexto al final de la primera subsección en paralelo y reanudar la decodificación de entropía adaptable al contexto de nuevo al comienzo de cualquier subsección paralela siguiente incluyendo, en el primer modo, una inicialización de las probabilidades de símbolo dependiente de los estados guardados de probabilidades de símbolo de la subsección paralela precedente y en el segundo modo, una inicialización de las probabilidades de símbolo independientemente de cualquier sector previamente decodificado y cualquier subsección en paralelo previamente decodificada.

Por lo tanto, la descripción anterior reveló métodos para la codificación, decodificación, encapsulación y transmisión con bajo retardo de datos de video estructurados tal como se proporcionan por el nuevo estándar de codificación de HEVC, tal como estructurado en mosaicos, subflujos de procesamiento en paralelo de frente de onda (WPP), sectores o sectores de entropía.

Especialmente, se ha definido cómo transportar datos codificados en paralelo en un escenario conversacional con el fin de ganar latencia mínima en el proceso de codificación, decodificación y transmisión. Por consiguiente, se ha descrito un procedimiento de codificación, transmisión y decodificación en paralelo de tubería con el fin de permitir aplicaciones de retardo mínimo como juegos, cirugía a distancia, etc.

Además, las realizaciones anteriores cerraron el espacio del procesamiento en paralelo de frente de onda (WPP) para hacerlo utilizable en escenarios de transmisión con bajo retardo. Por consiguiente, se ha presentado un nuevo formato de encapsulación para subflujos 0 de WPP, un sector dependiente. Este sector dependiente puede contener datos de sector de entropía, un subflujo de WPP, una fila completa de LCU, solo un fragmento de un sector, en donde el encabezamiento de sector transmitido previo también se aplica a los datos de fragmento contenidos. Los datos contenidos se señalan en el encabezamiento de subsector.

Se observará finalmente, que la nomenclatura para los nuevos sectores podría también ser "subconjunto / sectores de peso ligero", pero el nombre "sector dependiente" se ha encontrado que es mejor.

Se ha presentado una señalización que describe el nivel de paralelización en codificación y transporte.

Se describirán realizaciones adicionales y aspectos de la invención que pueden usarse de manera individual o en combinación con las características y funcionalidades descritas en el presente documento.

Según un primer aspecto es un codificador para reconstruir una instantánea 10 a partir de un flujo de datos 12 en el cual se codifica la instantánea en unidades de sectores 14 en las cuales se divide la instantánea 10 está configurado para decodificar los sectores 14 del flujo de datos 12 según un orden de sector 16 y el decodificador responde a una

- porción de elemento de sintaxis 18 dentro de un sector actual de los sectores, para decodificar el sector actual según al menos uno de al menos dos modos 20, 22, y según uno primero 20 de los al menos dos modos, decodificar el sector actual del flujo de datos 12 utilizando decodificación de entropía adaptable al contexto 24 que incluye una derivación de contextos a través de límites de sector, una actualización continua de probabilidades de símbolo de los contextos y una inicialización 38, 40 de las probabilidades de símbolo dependiente de los estados guardados de las probabilidades de símbolo de un sector previamente decodificado, y decodificación predictiva a través de los límites de sector, y según uno segundo 22 de los al menos dos modos, decodificar el sector actual del flujo de datos 12 utilizando decodificación de entropía adaptable al contexto con restricción de la derivación de los contextos para no cruzar los límites de sector, una actualización continua de probabilidades de símbolo de los contextos y una inicialización de las probabilidades de símbolo independientemente de cualquier sector previamente decodificado, y decodificación predictiva con restricción de la decodificación predictiva para no cruzar los límites de sector,
- Según un segundo aspecto cuando hace referencia de nuevo al primer aspecto, la instantánea 10 se divide en bloques de codificación 32 dispuestos en filas y columnas y que tienen un orden de exploración de trama 36 definido entre sí y el decodificador está configurado para asociar cada sector 14 con un subconjunto continuo de los bloques de codificación 32 en el orden de exploración de trama 36 de tal manera que los subconjuntos siguen entre sí a lo largo del orden de exploración de trama 36 según el orden de sector.
- Según un tercer aspecto cuando hace referencia de nuevo al segundo aspecto, el decodificador está configurado para guardar probabilidades de símbolo tal como se obtienen en la decodificación de entropía adaptable al contexto del sector previamente decodificado a un segundo bloque de codificación 32 en una fila según el orden de exploración de trama 36 y en la inicialización de las probabilidades de símbolo para la decodificación de entropía adaptable al contexto del sector actual según el primer modo, verificar si un primer bloque de codificación del subconjunto continuo de bloques de codificación 32 asociados con el sector actual es un primer bloque de codificación 32 en una fila según el orden de exploración de trama y, si es así, inicializar 40 las probabilidades de símbolo para la decodificación de entropía adaptable al contexto del sector actual dependiendo de las probabilidades de símbolo guardadas, tal como se obtienen en la decodificación de entropía adaptable al contexto del sector previamente decodificado hasta un segundo bloque de codificación en una fila según el orden de exploración de trama 36, y si no, inicializar 38 las probabilidades de símbolo para la decodificación de entropía adaptable al contexto del sector actual dependiendo de las probabilidades de símbolo, tal como se obtienen en decodificación de entropía adaptable al contexto del sector previamente decodificado hasta el final del sector previamente decodificado.
- Según un cuarto aspecto cuando hace referencia de nuevo a los aspectos primero a tercero, en el que el decodificador está configurado para responder a la porción de elemento de sintaxis 18 dentro del sector actual de los sectores 14, para decodificar el sector actual según uno de al menos tres modos, concretamente en uno del primero 20 y un tercer modo 42 o un segundo modo 22, en el que el decodificador está configurado para según el tercer modo 42 decodificar el sector actual del flujo de datos utilizando decodificación de entropía adaptable al contexto con restricción de la derivación de los contextos para no cruzar los límites de sector, una actualización continua de probabilidades de símbolo de los contextos y una inicialización de las probabilidades de símbolo independientemente de cualquier sector previamente decodificado y decodificación predictiva a través de los límites de sector, en el que uno del primer y tercer modos se selecciona dependiendo de un elemento de sintaxis.
- Según un quinto aspecto cuando hace referencia de nuevo a los aspectos primero a tercero, el decodificador está configurado para responder a un elemento de sintaxis genérico en el flujo de datos para operar en uno de al menos dos modos de operación genéricos con, según un primer modo de operación genérico, efectuar la capacidad de respuesta a la porción de elementos de sintaxis para cada sector, y, según un segundo modo de operación genérico, inevitablemente utilizar uno diferente de los al menos dos modos distintos del primer modo.
- Según un sexto aspecto cuando hace referencia de nuevo al segundo aspecto, el decodificador está configurado para, según el primer y segundo modo, inevitablemente y de manera ininterrumpida continuar actualizando continuamente las probabilidades de símbolo desde un comienzo hasta el fin del sector actual.
- Según un séptimo aspecto cuando hace referencia de nuevo al segundo aspecto, el decodificador está configurado para guardar probabilidades de símbolo tal como se obtienen en la decodificación de entropía adaptable al contexto del sector previamente decodificado hasta un final del sector previamente decodificado, y, al inicializar las probabilidades de símbolo para la codificación de entropía adaptable al contexto del sector actual según el primer modo, inicializar las probabilidades de símbolo para la codificación de entropía adaptable al contexto del sector actual dependiendo de las probabilidades de símbolo guardadas.
- Según un octavo aspecto cuando hace referencia de nuevo a los aspectos primero a séptimo, el decodificador está configurado para, en el primer y segundo modo, restringir la decodificación predictiva dentro de mosaicos en los que se subdivide la instantánea.

- Según un noveno aspecto cuando hace referencia de nuevo a los aspectos primero a octavo, el decodificador está configurado para, en el primer y segundo modo, leer información del sector actual que revela una subdivisión del sector actual en sub-secciones en paralelo, detener la decodificación de entropía adaptable al contexto al final de la primera subsección en paralelo y reanudar la decodificación de entropía adaptable al contexto de nuevo en un comienzo de cualquier subsección paralela siguiente incluyendo, en el primer modo, una inicialización de las probabilidades de símbolo dependiendo de los estados guardados de probabilidades de símbolo de la subsección paralela precedente y en el segundo modo, una inicialización de las probabilidades de símbolo independientemente de cualquier sector previamente decodificado y cualquier subsección paralela previamente decodificada.
- Según un décimo aspecto cuando hace referencia de nuevo a los aspectos primero a noveno, el decodificador está configurado para, según el primero 20 de los al menos dos modos, copiar para el sector actual una porción de una sintaxis de encabezamiento de sector de un sector precedente decodificado en el segundo modo.
- Según un decimoprimer aspecto cuando hace referencia de nuevo a los aspectos primero a décimo, el decodificador está configurado para reconstruir la instantánea 10 a partir del flujo de datos 12 utilizando procesamiento de WPP, en el que cada sector 14 comprende una porción de sintaxis de inicio 400 que indica una posición de un comienzo de decodificación del sector respectivo dentro de la instantánea 10 y en el que el decodificador está configurado para identificar puntos de entrada de subflujos de WPP en los que se agrupan los sectores, identificando, utilizando las porciones de sintaxis de inicio de sector, sectores que inician en el lado izquierdo de la instantánea, y decodificación en paralelo de los subflujos de WPP de manera escalonada comenzando secuencialmente la decodificación de los subflujos de WPP según el orden de sector.
- Según un decimosegundo aspecto, un codificador para codificar una instantánea 10 en un flujo de datos 12 en unidades de sectores 14 en las que se divide la instantánea 10, en el que el codificador está configurado para codificar los sectores 14 en el flujo de datos 12 según un orden de sector 16 y el codificador está configurado para determinar una porción de elemento de sintaxis 18, y codificar la misma en un sector actual de los sectores de tal manera que la porción de elemento de sintaxis señala el sector actual a codificarse según uno de al menos dos modos 20, 22, y si el sector actual se ha de codificar según uno primero 20 de los al menos dos modos, codificar el sector actual en el flujo de datos 12 utilizando codificación de entropía adaptable al contexto 24 incluyendo una derivación de contextos a través de límites de sector, una actualización continua de probabilidades de símbolo de los contextos y una inicialización 38, 40 de las probabilidades de símbolo dependiendo de los estados guardados de probabilidades de símbolo de un sector previamente codificado, y codificación predictiva a través de los límites de sector, y si el sector actual se ha de codificar según uno segundo 22 de los al menos dos modos, codificar el sector actual en el flujo de datos 12 utilizando codificación de entropía adaptable al contexto con restricción de la derivación de los contextos para no cruzar los límites de sector, una actualización continua de probabilidades de símbolo de los contextos y una inicialización de las probabilidades de símbolo independientemente de cualquier sector previamente codificado, y codificación predictiva con restricción de la codificación predictiva para no cruzar los límites de sector.
- Según un decimotercer aspecto cuando hace referencia de nuevo al decimosegundo aspecto, la instantánea 10 se divide en bloques de codificación 32 dispuestos en filas y columnas y que tienen un orden de exploración de trama 36 definido entre sí, y el codificador está configurado para asociar cada sector 14 con un subconjunto continuo de los bloques de codificación 32 en el orden de exploración de trama 36 de tal manera que los subconjuntos se siguen entre sí a lo largo del orden de exploración de trama 36 según el orden de sector.
- Según un decimocuarto aspecto cuando hace referencia de nuevo al decimotercer aspecto, el codificador está configurado para guardar probabilidades de símbolo tal como se obtienen en la codificación de entropía adaptable al contexto del sector previamente codificado hasta un segundo bloque de codificación 32 en una fila según el orden de exploración de trama 36 y en la inicialización de las probabilidades de símbolo para la codificación de entropía adaptable al contexto del sector actual según el primer modo, verificar si un primer bloque de codificación del subconjunto continuo de bloques de codificación 32 asociados con el sector actual es un primer bloque de codificación 32 en una fila según el orden de exploración de trama y si es así, inicializar 40 las probabilidades de símbolo para la codificación de entropía adaptable al contexto del sector actual dependiendo de las probabilidades de símbolo guardadas tal como se obtienen en la codificación de entropía adaptable al contexto del sector previamente codificado a un segundo bloque de codificación en una fila según el orden de exploración de trama 36 y si no, inicializar 38 las probabilidades de símbolo para la codificación de entropía adaptable al contexto del sector actual dependiendo de las probabilidades de símbolo tal como se obtienen en la codificación de entropía adaptable al contexto del sector previamente decodificado hasta el final del sector previamente codificado.
- Según un decimoquinto aspecto cuando hace referencia de nuevo a los aspectos decimosegundo a decimocuarto, el codificador está configurado para codificar la porción de elemento de sintaxis 18 en el sector actual de los sectores 14 de tal manera que el sector actual se señala para codificarse en el mismo según uno de al menos tres modos, concretamente en uno del primero 20 y un tercer modo 42 o un segundo modo 22, en el que el codificador está configurado para según el tercer modo 42, codificar el sector actual en el flujo de datos utilizando codificación de entropía adaptable al contexto con restricción de la derivación de los contextos para no cruzar los límites de sector,

una actualización continua de probabilidades de símbolo de los contextos y una inicialización de las probabilidades de símbolo independientemente de cualquier sector previamente codificado y codificación predictiva a través de los límites de sector, en el que el codificador distingue entre uno de los modos primero y tercero utilizando un elemento de sintaxis.

5

Según un decimosexto aspecto cuando hace referencia de nuevo a los aspectos decimosegundo a decimoquinto, el codificador está configurado para determinar un elemento de sintaxis genérico y escribir el mismo en el flujo de datos con operación en uno de al menos dos modos de operación genéricos dependiendo del elemento de sintaxis genérico, concretamente, con, según un primer modo de operación genérico, efectuar la codificación de la porción de elemento de sintaxis para cada sector, y, según un segundo modo de operación genérico, inevitablemente utilizar uno diferente de los al menos dos modos distintos del primer modo.

10

Según un decimoséptimo aspecto cuando hace referencia de nuevo al decimotercer aspecto, el codificador está configurado para, según los modos primero y segundo, inevitablemente y de manera ininterrumpida continuar la actualización continua de las probabilidades de símbolo desde el comienzo hasta el final del sector actual

15

Según un decimoctavo aspecto cuando hace referencia de nuevo al decimotercer aspecto, el decodificador está configurado para guardar probabilidades de símbolo tal como se obtienen en la decodificación de entropía adaptable al contexto del sector previamente decodificado hasta un final del sector previamente decodificado, y, al inicializar las probabilidades de símbolo para la codificación de entropía adaptable al contexto del sector actual según el primer modo, inicializar las probabilidades de símbolo para la codificación de entropía adaptable al contexto del sector actual dependiendo de las probabilidades de símbolo guardadas.

20

Según un decimonoveno aspecto cuando hace referencia de nuevo a los aspectos duodécimo a decimoctavo, el decodificador está configurado para, en el primer y segundo modo, restringir la decodificación predictiva dentro de mosaicos en los que se subdivide la instantánea.

25

Según un vigésimo aspecto, un decodificador para reconstruir una instantánea 10 a partir de un flujo de datos 12 en el que la instantánea se codifica en unidades de sectores 14 en las que se divide la instantánea 10, usando procesamiento de WPP, está configurado para decodificar los sectores 14 a partir del flujo de datos 12 según un orden de sector 16, en el que cada sector 14 comprende una porción de sintaxis de inicio 400 que indica una posición de un comienzo de decodificación del sector respectivo dentro de la instantánea 10 y en el que el decodificador está configurado para identificar puntos de entrada de subflujos de WPP en los que se agrupan los sectores, identificando, utilizando las porciones de sintaxis de inicio de sector, sectores que inician en el lado izquierdo de la instantánea, y decodificación en paralelo de los subflujos de WPP de manera escalonada comenzando secuencialmente la decodificación de los subflujos de WPP según el orden de sector.

30

35

Según un vigésimo primer aspecto, un codificador para codificar una instantánea 10 en un flujo de datos 12 en el que la instantánea se codifica en unidades de sectores 14 en las que se divide la instantánea 10, usando procesamiento de WPP, está configurado para codificar los sectores 14 en el flujo de datos 12 según un orden de sector 16, en el que el codificador está configurado para proporcionar cada sector 14 con una porción de sintaxis de inicio 400 que indica una posición del comienzo de codificación del sector respectivo dentro de la instantánea 10 y agruparía los sectores en subflujos de WPP de tal manera que para cada subflujo de WPP, el primer sector en orden de sectores se inicia en el lado izquierdo de la instantánea, codifica en paralelo los subflujos de WPP de manera escalonada con el comienzo secuencialmente de la codificación de los subflujos de WPP según el orden de sector.

40

45

Según un vigésimo segundo aspecto, un método para reconstruir una instantánea 10 a partir de un flujo de datos 12 en la que se codifica la instantánea en unidades de sectores 14 en las que se divide la instantánea 10, en el que el método comprende decodificar los sectores 14 del flujo de datos 12 según un orden de sector 16 y el método responde a una porción de elemento de sintaxis 18 dentro de un sector actual de los sectores, para decodificar el sector actual según uno de al menos dos modos 20, 22, en el que según uno primero 20 de los al menos dos modos, el sector actual se decodifica a partir del flujo de datos 12 utilizando decodificación de entropía adaptable al contexto 24 que incluye una derivación de contextos a través de límites de sector, una actualización continua de probabilidades de símbolo de los contextos y una inicialización 38, 40 de las probabilidades de símbolo dependiendo de los estados guardados de probabilidades de símbolo de un sector previamente decodificado, y decodificación predictiva a través de los límites de sector, y según uno segundo 22 de los al menos dos modos, el sector actual se decodifica del flujo de datos 12 utilizando decodificación de entropía adaptable al contexto con restricción de la derivación de los contextos para no cruzar los límites de sector, una actualización continua de probabilidades de símbolo de los contextos y una inicialización de las probabilidades de símbolo independientemente de cualquier sector previamente decodificado, y decodificación predictiva con restricción de la decodificación predictiva para no cruzar los límites de sector,

50

55

60

Según un vigésimo tercer aspecto, un método para codificar una instantánea 10 en un flujo de datos 12 en unidades de sectores 14 en las que se divide la instantánea 10, en el que el método comprende codificar los sectores 14 en el flujo de datos 12 según un orden de sector 16 y el método comprende determinar una porción de elemento de sintaxis

18, y codificar la misma en un sector actual de los sectores de tal manera que la porción de elemento de sintaxis señala el sector actual a codificarse según uno de al menos dos modos 20, 22, y si el sector actual se ha de codificar según uno primero 20 de los al menos dos modos, codificar el sector actual en el flujo de datos 12 utilizando codificación de entropía adaptable al contexto 24 incluyendo una derivación de contextos a través de límites de sector, una actualización continua de probabilidades de símbolo de los contextos y una inicialización 38, 40 de las probabilidades de símbolo dependiendo de los estados guardados de probabilidades de símbolo de un sector previamente codificado, y codificación predictiva a través de los límites de sector, y si el sector actual se ha de codificar según uno segundo 22 de los al menos dos modos, codificar el sector actual en el flujo de datos 12 utilizando codificación de entropía adaptable al contexto con restricción de la derivación de los contextos para no cruzar los límites de sector, una actualización continua de probabilidades de símbolo de los contextos y una inicialización de las probabilidades de símbolo independientemente de cualquier sector previamente codificado, y codificación predictiva con restricción de la codificación predictiva para no cruzar los límites de sector.

Según un vigésimo cuarto aspecto, un método para reconstruir una instantánea 10 a partir de un flujo de datos 12 en la que se codifica la instantánea en unidades de sectores 14 en las que se divide la instantánea 10, usando procesamiento de WPP, decodificar los sectores 14 del flujo de datos 12 según un orden de sector 16, en el que cada sector 14 comprende una porción de sintaxis de inicio 400 que indica una posición del comienzo de codificación del sector respectivo dentro de la instantánea 10 y el método comprende además identificar puntos de entrada de subflujos de WPP en los que se agrupan los sectores, identificando, utilizando las porciones de sintaxis de inicio de sector, sectores que inician en el lado izquierdo de la instantánea, y decodificación en paralelo de los subflujos de WPP de manera escalonada comenzando secuencialmente la decodificación de los subflujos de WPP según el orden de sector.

Según un vigésimo quinto aspecto, un método para codificar una instantánea 10 en un flujo de datos 12 en unidades de sectores 14 en las que se divide la instantánea 10, usando procesamiento de WPP, comprende codificar los sectores 14 en el flujo de datos 12 según un orden de sector 16, y proporcionar cada sector 14 con una porción de sintaxis de inicio 400 que indica una posición del comienzo de codificación del sector respectivo dentro de la instantánea 10 y agrupar adicionalmente los sectores en subflujos de WPP de tal manera que para cada subflujo de WPP, el primer sector en orden de sectores se inicia en el lado izquierdo de la instantánea, codifica en paralelo los subflujos de WPP de manera escalonada con el comienzo secuencialmente de la codificación de los subflujos de WPP según el orden de sector

Un aspecto vigésimo sexto se refiere a un programa informático que tiene un código de programa para realizar, cuando se ejecuta en un ordenador, un método según los aspectos vigésimo segundo o vigésimo quinto aspecto.

Aunque algunos aspectos han sido descritos en el contexto de un aparato, es evidente que estos aspectos también representan una descripción del método correspondiente, en donde un bloque o dispositivo corresponde a una etapa de método o un aspecto de una etapa de método. Análogamente, los aspectos descritos en el contexto de una etapa de método también representan una descripción de un bloque o elemento o característica de un aparato correspondiente. Algunas o todas las etapas de método pueden ejecutarse por (o usando) un aparato de hardware, como por ejemplo, un microprocesador, un ordenador programable o un circuito electrónico. En algunas realizaciones, alguna o más de las etapas de método más importantes pueden ejecutarse mediante tal aparato.

Dependiendo de ciertos requisitos de implementación, las realizaciones de la invención pueden implementarse en hardware o en software. La implementación puede realizarse utilizando un medio de almacenamiento digital, por ejemplo un disco flexible, un DVD, un Blu-Ray, un CD, una ROM, una PROM, una EPROM, una EEPROM o una memoria FLASH, que tiene señales de control que se pueden leer electrónicamente almacenadas en la misma, que cooperan (o que son aptas para cooperar) con un sistema de ordenador programable, de tal manera que se efectúa el método respectivo. Por consiguiente, el medio de almacenamiento digital se puede leer por ordenador.

Algunas realizaciones según la invención comprenden un portador de datos que tiene señales de control que se pueden leer electrónicamente que son aptas para cooperar con un sistema de ordenador programable, de tal manera que se efectúa uno de los métodos descritos en el presente documento.

En general, las realizaciones de la presente invención pueden implementarse como un producto de programa de ordenador con códigos de programa, los códigos de programa pueden funcionar para efectuar uno de los métodos cuando el producto de programa de ordenador se ejecuta en un ordenador. Los códigos de programa pueden por ejemplo almacenarse en un portador que se puede leer por la máquina.

Otras realizaciones comprenden el programa de ordenador para efectuar uno de los métodos descritos en el presente documento, almacenado en un portador que se puede leer por la máquina.

En otras palabras, una realización del método de la invención es por consiguiente, un programa de ordenador que tiene códigos de programa para efectuar uno de los métodos descritos en el presente documento, cuando el programa de ordenador se ejecuta en un ordenador.

Una realización adicional de los métodos de la invención es por consiguiente, un portador de datos (o medio de almacenamiento digital o medio que se puede leer por ordenador) que comprende, registrado en el mismo, el programa de ordenador para efectuar uno de los métodos descritos en el presente documento. El portador de datos, medio de almacenamiento digital o el medio registrado son típicamente tangibles y/o no transitorios.

Una realización adicional del método de la invención es por consiguiente, un flujo de datos o una secuencia de señales que representan el programa de ordenador para efectuar uno de los métodos descritos en el presente documento. El flujo de datos o la secuencia de señales puede por ejemplo estar configurada para ser transferida mediante una conexión de comunicación de datos, por ejemplo, mediante Internet.

Una realización adicional comprende un medio de procesamiento, por ejemplo un ordenador o un dispositivo lógico programable, configurado para o adaptado para efectuar uno de los métodos descritos en el presente documento.

Una realización adicional comprende un ordenador que tiene instalado en el mismo el programa de ordenador para efectuar uno de los métodos descritos en el presente documento.

Una realización adicional según la invención comprende un aparato o un sistema configurado para transferir (por ejemplo, electrónicamente u ópticamente) un programa de ordenador para efectuar uno de los métodos descritos en el presente documento a un receptor. El receptor puede ser por ejemplo un ordenador, un dispositivo móvil, un dispositivo de memoria o similares. El aparato o sistema puede por ejemplo comprender un servidor de archivos para transferir el programa de ordenador al receptor.

En algunas realizaciones, se puede usar un dispositivo lógico programable (por ejemplo, una matriz de compuertas programables en el campo) para efectuar algunas o todas las funcionalidades de los métodos descritos en el presente documento. En algunas realizaciones, las matrices de compuertas programables en el campo pueden cooperar con un microprocesador con el fin de efectuar uno de los métodos descritos en el presente documento. En general, los métodos se efectúan preferentemente mediante cualquier aparato de hardware.

Las realizaciones descritas anteriormente son solamente ilustrativas para los principios de la presente invención. Se comprenderá que serán evidentes modificaciones y variaciones de las disposiciones y los detalles descritos en el presente documento para otros expertos en la materia. Se pretende por consiguiente limitarse solamente por el alcance de las siguientes reivindicaciones de patente y no por los detalles específicos presentados a modo de descripción y explicación de las realizaciones del presente documento.

Referencias

[1] Thomas Wiegand, Gary J. Sullivan, Gisle Bjontegaard, Ajay Luthra, "Overview of the H.264/AVC Video Coding Standard", IEEE Trans. Circuits Syst. Video Technol., vol. 13, N7, julio de 2003.

[2] JCT-VC, "High-Efficiency Video Coding (HEVC) text specification Working Draft 6", JCTVC-H1003, febrero de 2012.

[3] ISO/IEC 13818-1: MPEG-2 Systems specification

REIVINDICACIONES

1. Método para reconstruir una instantánea (10) a partir de un flujo de datos (12) en el cual se codifica la instantánea en unidades de sectores (14) en las cuales se divide la instantánea (10), en el que el método comprende decodificar los sectores (14) del flujo de datos (12) según una orden de sector (16) y el método responde a una porción de elemento de sintaxis (18) dentro de un sector actual de los sectores, para decodificar el sector actual según al menos uno de al menos dos modos (20, 22), en el que
 - según uno primero (20) de los al menos dos modos, el sector actual se decodifica del flujo de datos (12) utilizando decodificación de entropía adaptable al contexto (24) que incluye una derivación de contextos a través de límites de sector, una actualización continua de probabilidades de símbolo de los contextos y una inicialización (38, 40) de las probabilidades de símbolo dependiente de los estados guardados de las probabilidades de símbolo de un sector previamente decodificado, y decodificación predictiva a través de los límites de sector, y
 - según uno segundo (22) de los al menos dos modos, el sector actual se decodifica del flujo de datos (12) utilizando decodificación de entropía adaptable al contexto con restricción de la derivación de los contextos para no cruzar los límites de sector, una actualización continua de probabilidades de símbolo de los contextos y una inicialización de las probabilidades de símbolo independientemente de cualquier sector previamente decodificado, y decodificación predictiva con restricción de la decodificación predictiva para no cruzar los límites de sector,
 - en el que la instantánea (10) se divide en bloques de codificación (32) dispuestos en filas y columnas y que tienen un orden de exploración de trama (36) definido entre sí, y el método comprende asociar cada sector (14) con un subconjunto continuo de los bloques de codificación (32) en el orden de exploración de trama (36) de tal manera que los subconjuntos se siguen entre sí a lo largo del orden de exploración de trama (36) según el orden de sector, y
 - en el que el método comprende guardar probabilidades de símbolo tal como se obtienen en la decodificación de entropía adaptable al contexto del sector previamente decodificado a un segundo bloque de codificación (32) en una fila según el orden de exploración de trama (36) y, en la inicialización de las probabilidades de símbolo para la decodificación de entropía adaptable al contexto del sector actual según el primer modo, verificar si un primer bloque de codificación del subconjunto continuo de bloques de codificación (32) asociados con el sector actual es un primer bloque de codificación (32) en una fila según el orden de exploración de trama y, si es así, inicializar (40) las probabilidades de símbolo para la decodificación de entropía adaptable al contexto del sector actual dependiendo de las probabilidades de símbolo guardadas, tal como se obtienen en la decodificación de entropía adaptable al contexto del sector previamente decodificado hasta un segundo bloque de codificación en una fila según el orden de exploración de trama (36), y si no, inicializar (38) las probabilidades de símbolo para la decodificación de entropía adaptable al contexto del sector actual dependiendo de las probabilidades de símbolo, tal como se obtienen en decodificación de entropía adaptable al contexto del sector previamente decodificado hasta el final del sector previamente decodificado,
 - en el que el método comprende predicción de intra-fotograma y predicción de inter-fotograma.
2. Método según la reivindicación 1, que comprende dividir bloques de transformación de crominancia de manera diferente que los bloques de transformación de luminancia en respuesta a información en el flujo de datos.
3. Método según la reivindicación 1, en el que el método comprende, en respuesta a la porción de elemento de sintaxis (18) dentro del sector actual de los sectores (14), decodificando el sector actual según uno de al menos tres modos, concretamente en el uno del primero (20) y un tercer modo (42) o un segundo modo (22), en el que
 - según el tercer modo (42), el sector actual se decodifica del flujo de datos usando decodificación de entropía adaptable al contexto con restricción de la derivación de los contextos para no cruzar los límites de sector, una actualización continua de probabilidades de símbolo de los contextos y una inicialización de las probabilidades de símbolo independientemente de cualquier sector previamente decodificado, y decodificación predictiva a través de los límites de sector,
 - en el que el uno de los modos primero y tercero se selecciona dependiendo de un elemento de sintaxis.
4. Método para codificar una instantánea (10) en un flujo de datos (12) en unidades de sectores (14) en las que

se divide la instantánea (10), en el que el método comprende codificar los sectores (14) en el flujo de datos (12) según un orden de sector (16) y el método comprende

5 determinar una porción de elemento de sintaxis (18), y codificar la misma en un sector actual de los sectores de tal manera que la porción de elemento de sintaxis señala el sector actual a codificarse según uno de al menos dos modos (20, 22), y

10 si el sector actual se ha de codificar según uno primero (20) de los al menos dos modos, codificar el sector actual en el flujo de datos (12) utilizando codificación de entropía adaptable al contexto (24) incluyendo una derivación de contextos a través de límites de sector, una actualización continua de probabilidades de símbolo de los contextos y una inicialización (38, 40) de las probabilidades de símbolo dependiendo de los estados guardados de probabilidades de símbolo de un sector previamente codificado, y codificación predictiva a través de los límites de sector, y

15 si el sector actual se ha de codificar según uno segundo (22) de los al menos dos modos, codificar el sector actual en el flujo de datos (12) utilizando codificación de entropía adaptable al contexto con restricción de la derivación de los contextos para no cruzar los límites de sector, una actualización continua de probabilidades de símbolo de los contextos y una inicialización de las probabilidades de símbolo independientemente de cualquier sector previamente codificado, y codificación predictiva con restricción de la codificación predictiva para no cruzar los límites de sector,

20 en el que la instantánea (10) se divide en bloques de codificación (32) dispuestos en filas y columnas y que tienen un orden de exploración de trama (36) definido entre sí, y el método comprende asociar cada sector (14) con un subconjunto continuo de los bloques de codificación (32) en el orden de exploración de trama (36) de tal manera que los subconjuntos se siguen entre sí a lo largo del orden de exploración de trama (36) según el orden de sector, y

25 en el que el método comprende adicionalmente guardar probabilidades de símbolo tal como se obtienen en la codificación de entropía adaptable al contexto del sector previamente codificado hasta un segundo bloque de codificación (32) en una fila según el orden de exploración de trama (36) y en la inicialización de las probabilidades de símbolo para la codificación de entropía adaptable al contexto del sector actual según el primer modo, verificar si un primer bloque de codificación del subconjunto continuo de bloques de codificación (32) asociados con el sector actual es un primer bloque de codificación (32) en una fila según el orden de exploración de trama y si es así, inicializar (40) las probabilidades de símbolo para la codificación de entropía adaptable al contexto del sector actual dependiendo de las probabilidades de símbolo guardadas tal como se obtienen en la codificación de entropía adaptable al contexto del sector previamente codificado a un segundo bloque de codificación en una fila según el orden de exploración de trama (36) y si no, inicializar (38) las probabilidades de símbolo para la codificación de entropía adaptable al contexto del sector actual dependiendo de las probabilidades de símbolo tal como se obtienen en la codificación de entropía adaptable al contexto del sector previamente decodificado hasta el final del sector previamente codificado,

en el que el método comprende predicción de intra-fotograma y predicción de inter-fotograma.

45 5. Método según la reivindicación 4, que comprende dividir bloques de transformación de crominancia de manera diferente que los bloques de transformación de luminancia.

6. Flujo de datos codificado usando el método según la reivindicación 4.

50 7. Flujo de datos según la reivindicación 6, en el que los bloques de transformación de crominancia se dividen de manera diferente que los bloques de transformación de luminancia.

8. Programa informático que tiene un código de programa para realizar, cuando se ejecuta en un ordenador, un método según la reivindicación 1 o 4.

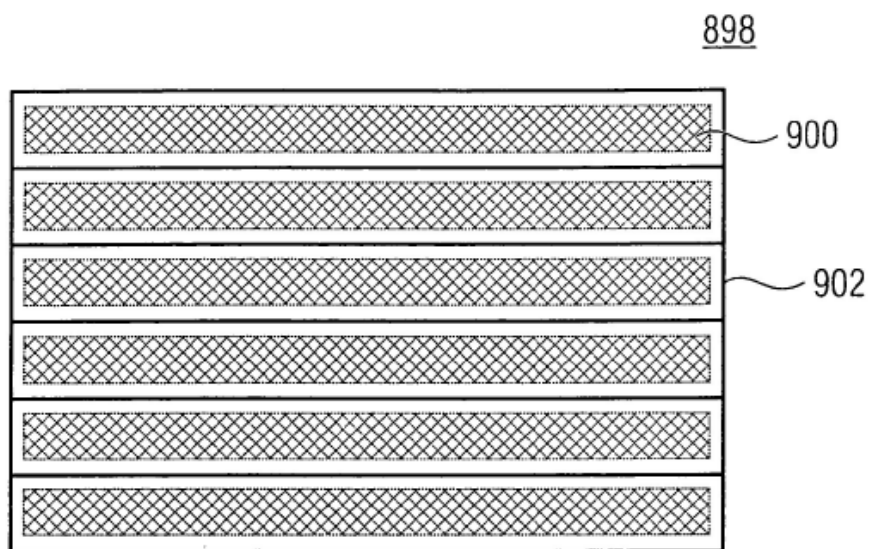


FIG 1

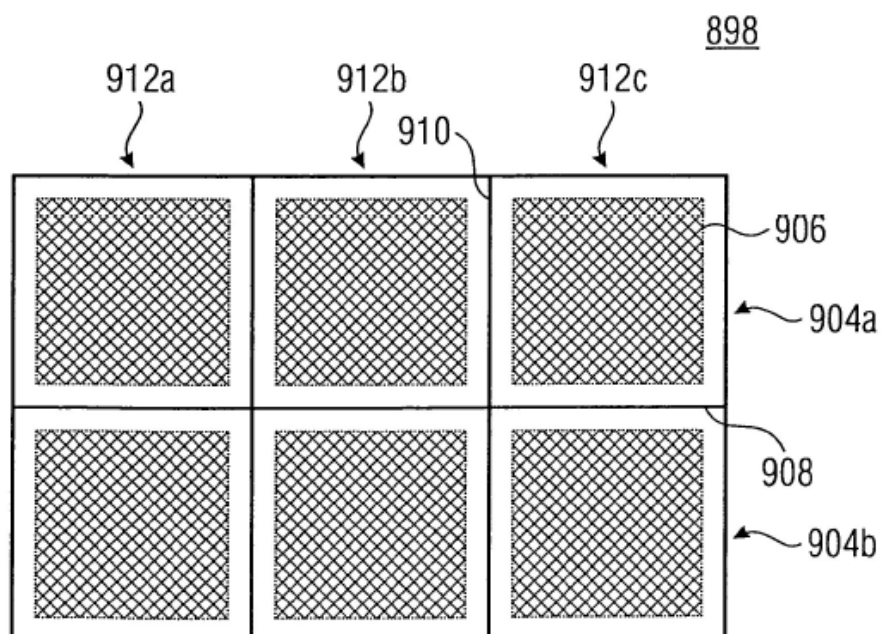


FIG 2

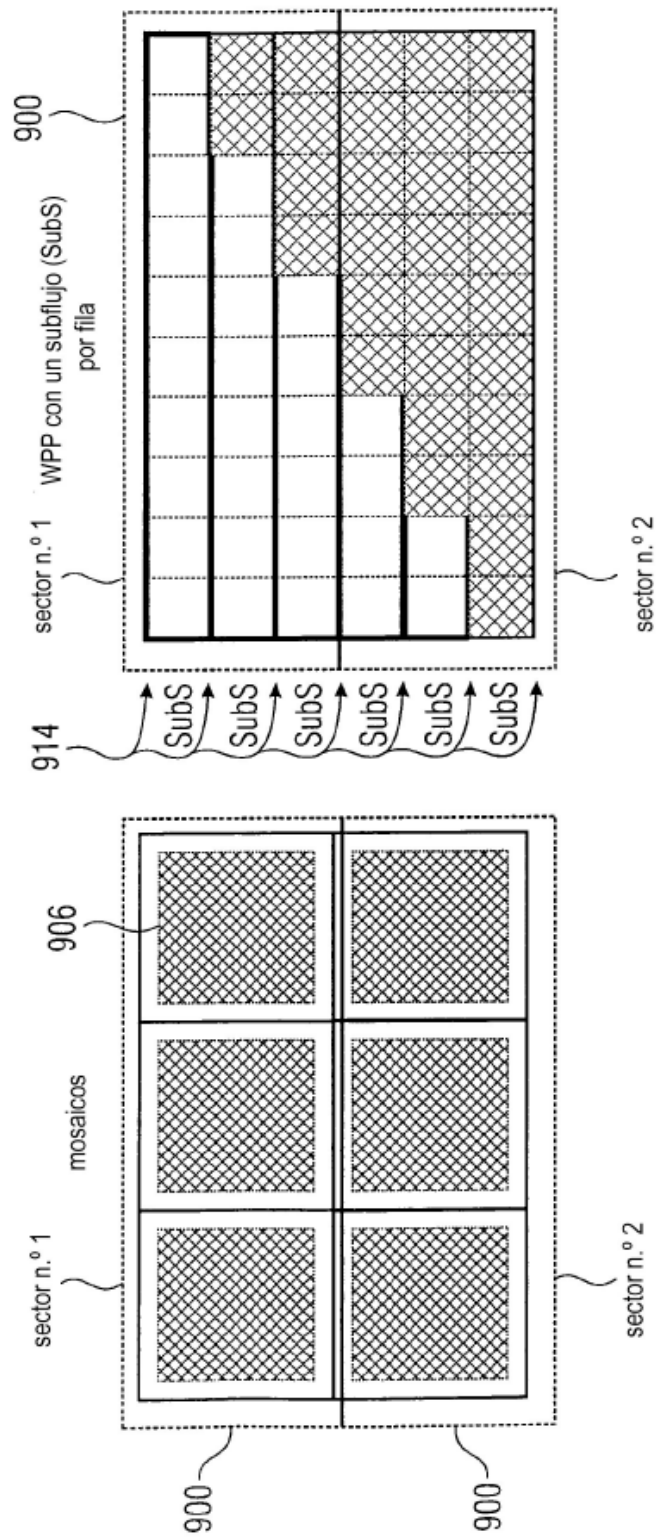


FIG 3

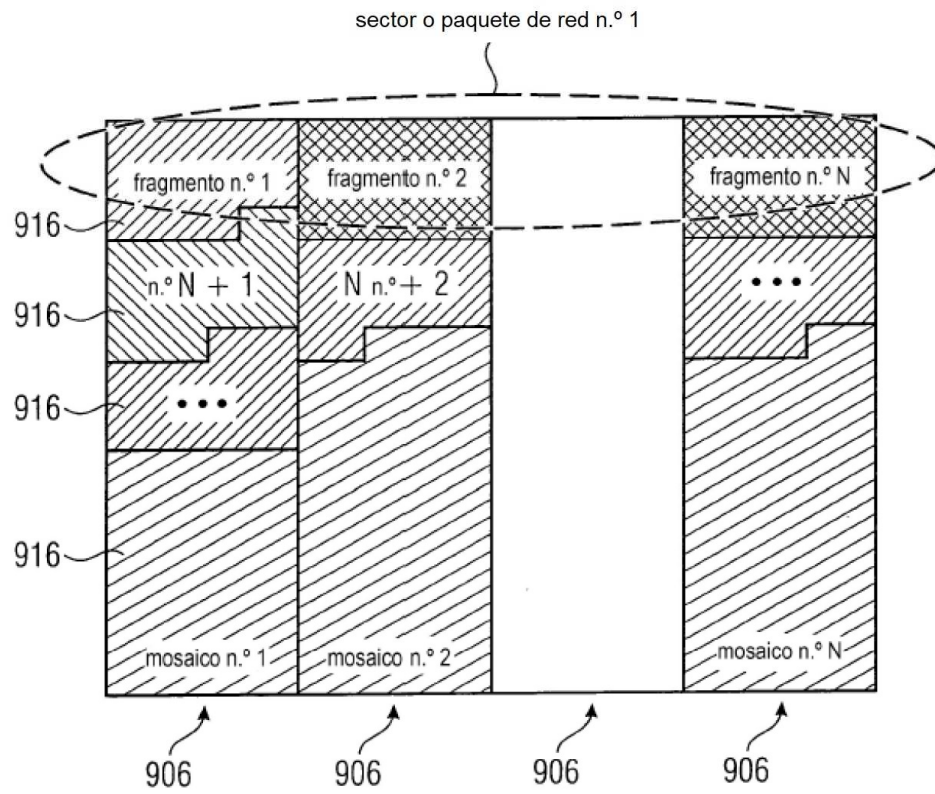


FIG 4

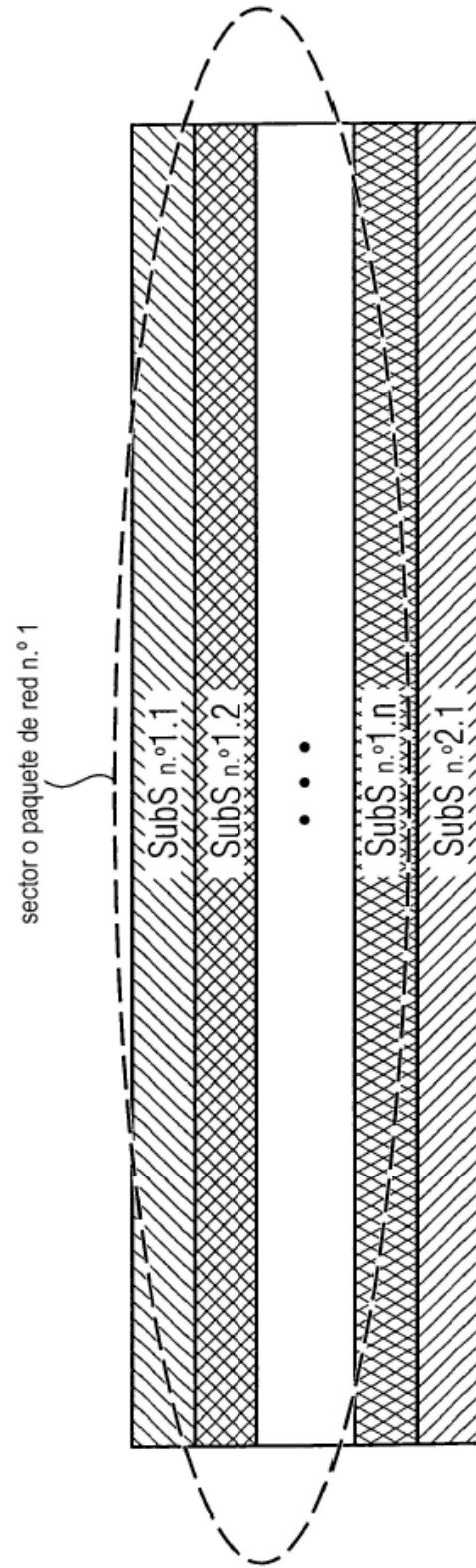


FIG 5

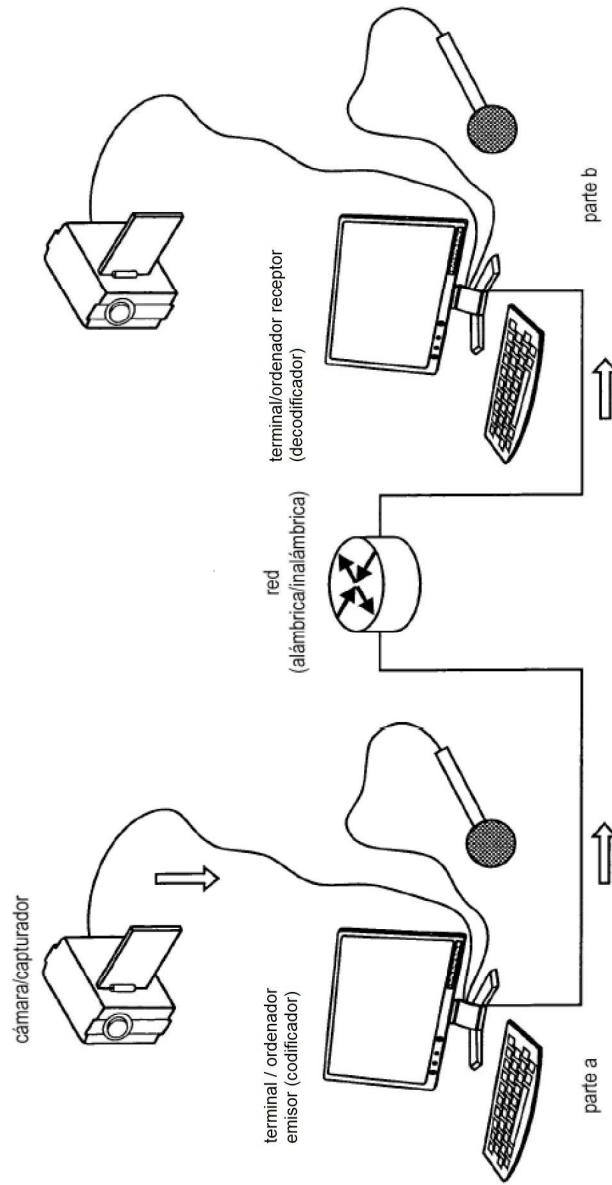


FIG 6

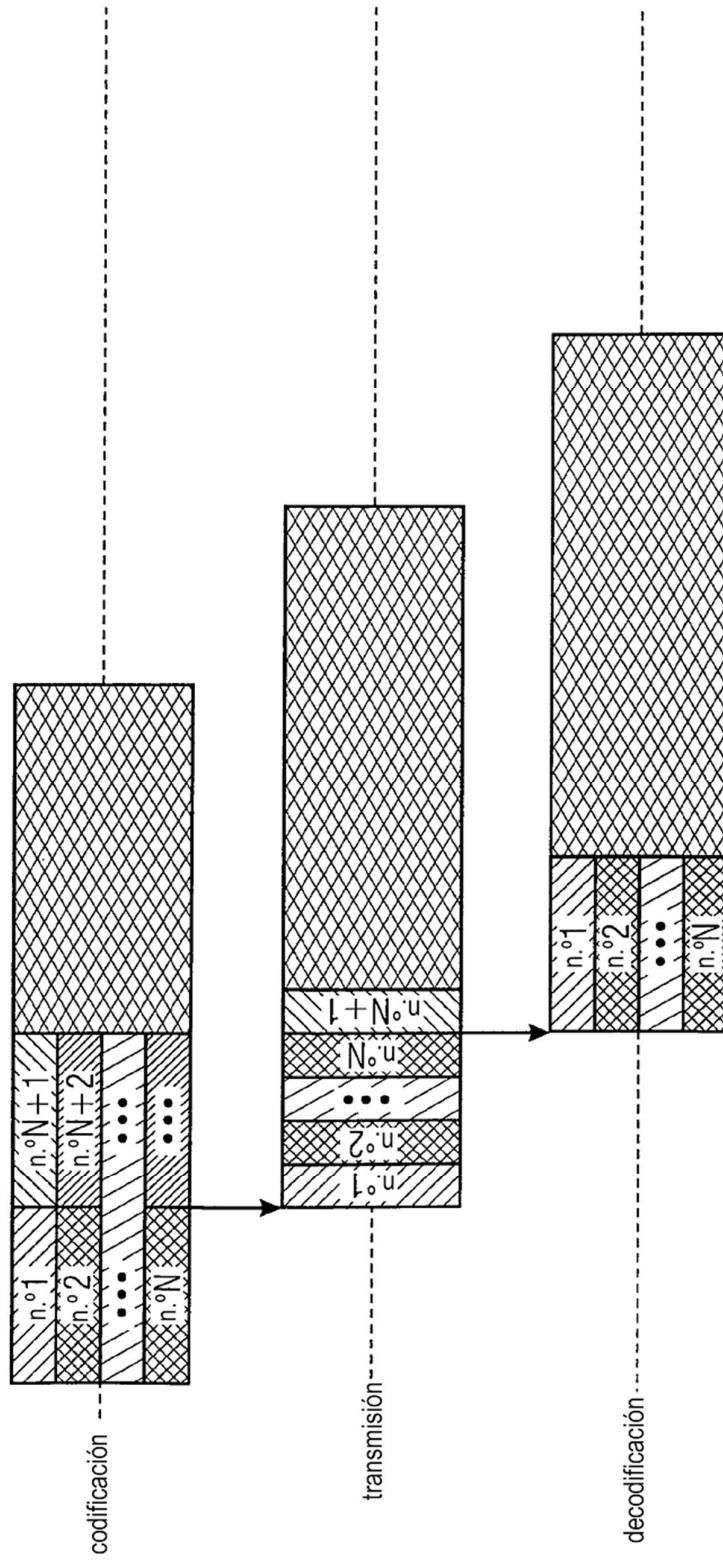


FIG 7

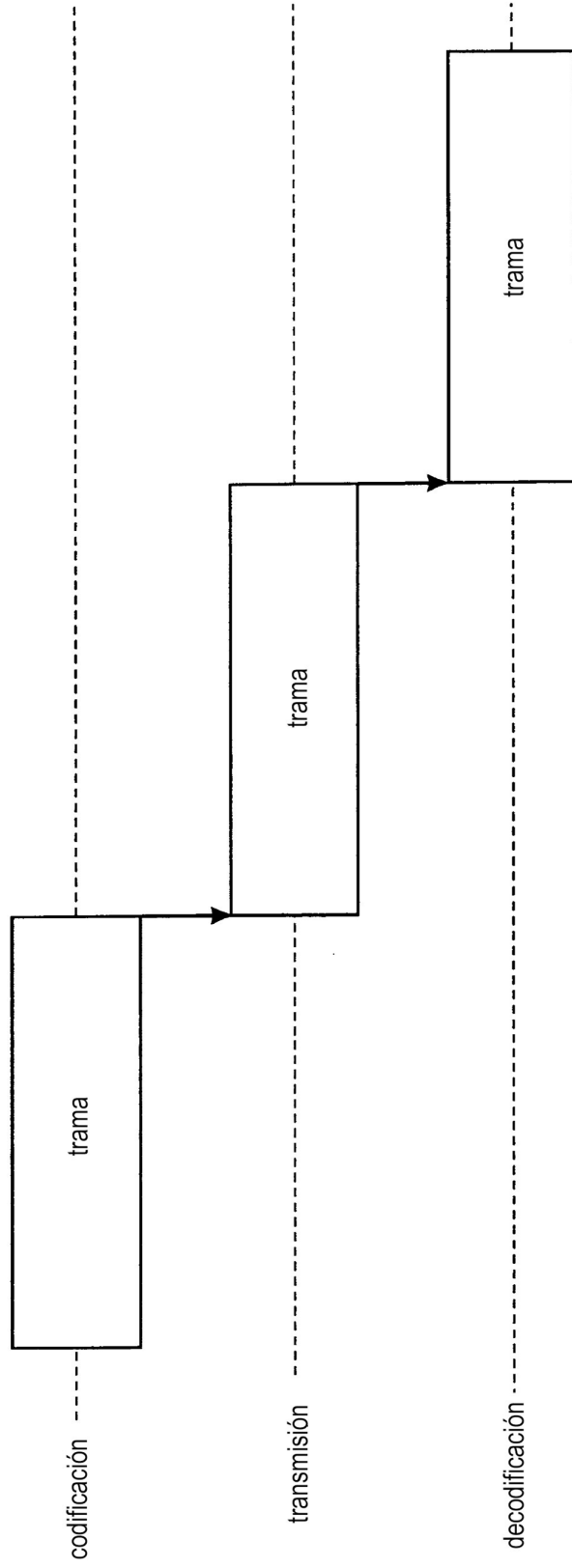


FIG 8

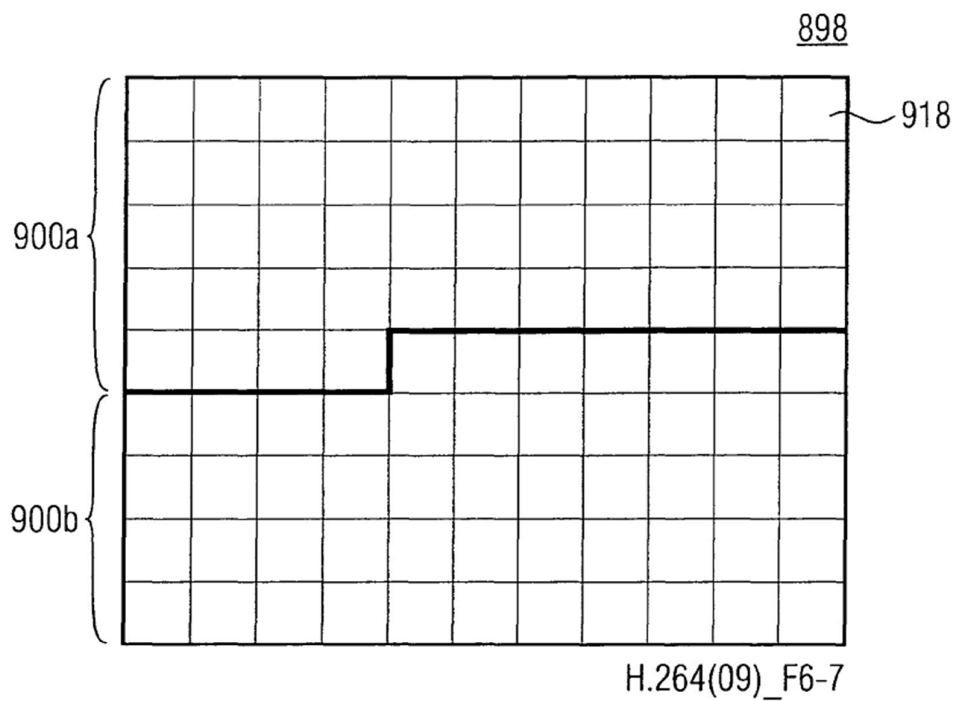


FIG 9

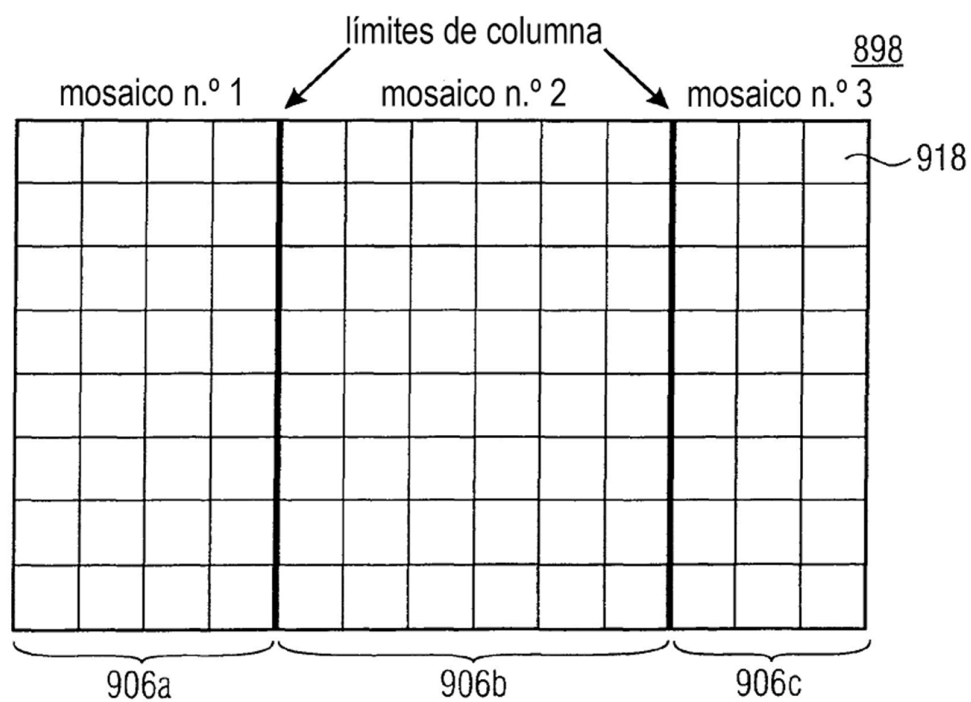


FIG 10

seq_parameter_set_rbsp() {	Descriptor
profile_idc	u(8)
reserved_zero_8bits /* igual a 0 */	u(8)
level_idc	u(8)
seq_parameter_set_id	ue(v)
chroma_format_idc	ue(v)
si(chroma_format_idc == 3)	
separate_colour_plane_flag	u(1)
max_temporal_layers_minus1	u(3)
pic_width_in_luma_samples	ue(v)
pic_height_in_luma_samples	ue(v)
pic_cropping_flag	u(1)
si(pic_cropping_flag) {	
pic_crop_left_offset	ue(v)
pic_crop_right_offset	ue(v)
pic_crop_top_offset	ue(v)
pic_crop_bottom_offset	ue(v)
}	
bit_depth_luma_minus8	ue(v)
bit_depth_chroma_minus8	ue(v)
pcm_enabled_flag	u(1)
si(pcm_enabled_flag) {	
pcm_bit_depth_luma_minus1	u(4)
pcm_bit_depth_chroma_minus1	u(4)
}	
qpprime_y_zero_transquant_bypass_flag	u(1)
log2_max_pic_order_cnt_lsb_minus4	ue(v)
para(i = 0; i <= max_temporal_layers_minus1; i++) {	
max_dec_pic_buffering[i]	ue(v)

FIG 11A

FIG 11	FIG 11A
	FIG 11B
	FIG 11C

num_reorder_pics[i]	ue(v)
max_latency_increase[i]	ue(v)
}	
restricted_ref_pic_lists_flag	u(1)
si (restricted_ref_pic_lists_flag)	
lists_modification_present_flag	u(1)
log2_min_coding_block_size_minus3	ue(v)
log2_diff_max_min_coding_block_size	ue(v)
log2_min_transform_block_size_minus2	ue(v)
log2_diff_max_min_transform_block_size	ue(v)
si (pcm_enabled_flag) {	
log2_min_pcm_coding_block_size_minus3	ue(v)
log2_diff_max_min_pcm_coding_block_size	ue(v)
}	
max_transform_hierarchy_depth_inter	ue(v)
max_transform_hierarchy_depth_intra	ue(v)
scaling_list_enable_flag	
chroma_pred_from_luma_enabled_flag	u(1)
deblocking_filter_in_aps_enabled_flag	u(1)
seq_loop_filter_across_slices_enabled_flag	u(1)
asymmetric_motion_partitions_enabled_flag	u(1)
non_square_quadtree_enabled_flag	u(1)
sample_adaptive_offset_enabled_flag	u(1)
adaptive_loop_filter_enabled_flag	u(1)
si (adaptive_loop_filter_enabled_flag)	
alf_coef_in_slice_flag	u(1)
si (pcm_enabled_flag)	
pcm_loop_filter_disable_flag	u(1)
temporal_id_nesting_flag	u(1)

FIG 11B

FIG 11	FIG 11A
	FIG 11B
	FIG 11C

si (log2_min_coding_block_size_minus3 == 0)	
inter_4x4_enabled_flag	u(1)
num_short_term_ref_pic_sets	ue(v)
para (i = 0; i < num_short_term_ref_pic_sets; i++)	
short_term_ref_pic_set(i)	
long_term_ref_pics_present_flag	u(1)
tiles_or_entropy_coding_sync_idc	u(2)
si (tiles_or_entropy_coding_sync_idc == 1) {	
num_tile_columns_minus1	ue(v)
num_tile_rows_minus1	ue(v)
uniform_spacing_flag	u(1)
si (!uniform_spacing_flag) {	
para (i = 0; i < num_tile_columns_minus1; i++)	
column_width[i]	ue(v)
para (i = 0; i < num_tile_rows_minus1; i++)	
row_height[i]	ue(v)
}	
loop_filter_across_tiles_enabled_flag	u(1)
}	
vui_parameters_present_flag	u(1)
si (vui_parameters_present_flag)	
vui_parameters()	
sps_extension_flag	u(1)
si (sps_extension_flag)	
mientras (more_rbsp_data())	
sps_extension_data_flag	u(1)
rbp_trailing_bits()	
}	

FIG 11C

FIG 11	FIG 11A
	FIG 11B
	FIG 11C

pic_parameter_set_rbsp() {	Descriptor
pic_parameter_set_id	ue(v)
seq_parameter_set_id	ue(v)
sign_data_hiding_flag	u(1)
si(sign_data_hiding_flag)	
sign_hiding_threshold	u(4)
cabac_init_present_flag	u(1)
num_ref_idx_l0_default_active_minus1	ue(v)
num_ref_idx_l1_default_active_minus1	ue(v)
pic_init_qp_minus26	se(v)
constrained_intra_pred_flag	u(1)
enable_temporal_mvp_flag	u(1)
slice_granularity	u(2)
max_cu_qp_delta_depth	ue(v)
cb_qp_offset	se(v)
cr_qp_offset	se(v)
weighted_pred_flag	u(1)
weighted_bipred_idc	u(2)
output_flag_present_flag	u(1)
si(tiles_or_entropy_coding_sync_idc == 1) {	
tile_info_present_flag	u(1)
tile_control_present_flag	u(1)
si(tile_info_present_flag) {	
num_tile_columns_minus1	ue(v)

FIG 12A

FIG 12	FIG 12A
	FIG 12B

num_tile_rows_minus1	ue(v)
uniform_spacing_flag	u(1)
si (!uniform_spacing_flag) {	
para (i = 0; i < num_tile_columns_minus1; i++)	
column_width[i]	ue(v)
para (i = 0; i < num_tile_rows_minus1; i++)	
row_height[i]	ue(v)
}	
}	
si (tile_control_present_flag)	
loop_filter_across_tiles_enabled_flag	u(1)
} si no si (tiles_or_entropy_coding_sync_idc == 2)	
num_substreams_minus1	ue(v)
deblocking_filter_control_present_flag	u(1)
si (slice_type == P slice_type == B)	
log2_parallel_merge_level_minus2	ue(v)
pps_extension_flag	u(1)
si (pps_extension_flag)	
mientras (more_rbsp_data())	
pps_extension_data_flag	u(1)
rbsp_trailing_bits()	
}	

FIG 12B

FIG 12	FIG 12A
	FIG 12B

slice_header() {	Descriptor
first_slice_in_pic_flag	u(1)
si (first_slice_in_pic_flag == 0)	
slice_address	u(v)
slice_type	ue(v)
entropy_slice_flag	u(1)
si (!entropy_slice_flag) {	
pic_parameter_set_id	ue(v)
si (output_flag_present_flag)	
pic_output_flag	u(1)
si (separate_colour_plane_flag == 1)	
colour_plane_id	u(2)
si (!drPicFlag) {	
idr_pic_id	ue(v)
no_output_of_prior_pics_flag	u(1)
} si no {	
pic_order_cnt_lsb	u(v)
short_term_ref_pic_set_sps_flag	u(1)
si (!short_term_ref_pic_set_sps_flag)	
short_term_ref_pic_set(num_short_term_ref_pic_sets)	
si no	
short_term_ref_pic_set_idx	u(v)
si (long_term_ref_pics_present_flag) {	
num_long_term_pics	ue(v)
para (i = 0; i < num_long_term_pics; i++) {	
delta_poc_lsb_lt[i]	ue(v)
delta_poc_msb_present_flag[i]	u(1)
si (delta_poc_msb_present_flag[i])	
delta_poc_msb_cycle_lt_minus1[i]	ue(v)
used_by_curr_pic_lt_flag[i]	u(1)
}	
}	
}	
si (sample_adaptive_offset_enabled_flag) {	
slice_sao_interleaving_flag	u(1)
slice_sample_adaptive_offset_flag	u(1)

FIG 13A

FIG 13	FIG 13A
	FIG 13B
	FIG 13C

si (slice_sao_interleaving_flag & & slice_sample_adaptive_offset_flag) {	
sao_cb_enable_flag	u(1)
sao_cr_enable_flag	u(1)
}	
}	
si (scaling_list_enable_flag deblocking_filter_in_aps_enabled_flag (sample_adaptive_offset_enabled_flag) & & !slice_sao_interleaving_flag) adaptive_loop_filter_enabled_flag)	
aps_id	ue(v)
si (slice_type == P slice_type == B) {	
num_ref_idx_active_override_flag	u(1)
si (num_ref_idx_active_override_flag) {	
num_ref_idx_l0_active_minus1	ue(v)
si (slice_type == B)	
num_ref_idx_l1_active_minus1	ue(v)
}	
}	
si (lists_modification_present_flag) {	
ref_pic_list_modification()	
ref_pic_list_combination()	
}	
si (slice_type == B)	
mvd_l1_zero_flag	u(1)
}	
si (cabac_init_present_flag & & slice_type != I)	
cabac_init_flag	u(1)
si (!entropy_slice_flag) {	
slice_qp_delta	se(v)
si (deblocking_filter_control_present_flag) {	
si (deblocking_filter_in_aps_enabled_flag)	
inherit_dbl_params_from_aps_flag	u(1)
si (!inherit_dbl_params_from_aps_flag) {	
disable_deblocking_filter_flag	u(1)
si (!disable_deblocking_filter_flag) {	
beta_offset_div2	se(v)

FIG 13B

FIG 13	FIG 13A
	FIG 13B
	FIG 13C

tc_offset_div2	se(v)
}	
}	
}	
si (slice_type == B)	
collocated_from_l0_flag	u(1)
si (slice_type != I && ((collocated_from_l0_flag && num_ref_idx_l0_active_minus1 > 0) (!collocated_from_l0_flag && num_ref_idx_l1_active_minus1 > 0))	
collocated_ref_idx	ue(v)
si ((weighted_pred_flag && slice_type == P) (weighted_bipred_idc == 1 && slice_type == B))	
pred_weight_table()	
}	
si (slice_type == P slice_type == B)	
five_minus_max_num_merge_cand	ue(v)
si (adaptive_loop_filter_enabled_flag) {	
slice_adaptive_loop_filter_flag	u(1)
si (slice_adaptive_loop_filter_flag && alf_coef_in_slice_flag)	
alf_param()	
si (slice_adaptive_loop_filter_flag && !alf_coef_in_slice_flag)	
alf_cu_control_param()	
}	
si (seq_loop_filter_across_slices_enabled_flag && (slice_adaptive_loop_filter_flag slice_sample_adaptive_offset_flag !disable_deblocking_filter_flag))	
slice_loop_filter_across_slices_enabled_flag	u(1)
si (tiles_or_entropy_coding_sync_idc > 0) {	
num_entry_point_offsets	ue(v)
si (num_entry_point_offsets > 0) {	
offset_len_minus1	ue(v)
para (i = 0; i < num_entry_point_offsets; i++)	
entry_point_offset[i]	u(v)
}	
}	
}	

FIG 13C

FIG 13	FIG 13A
	FIG 13B
	FIG 13C

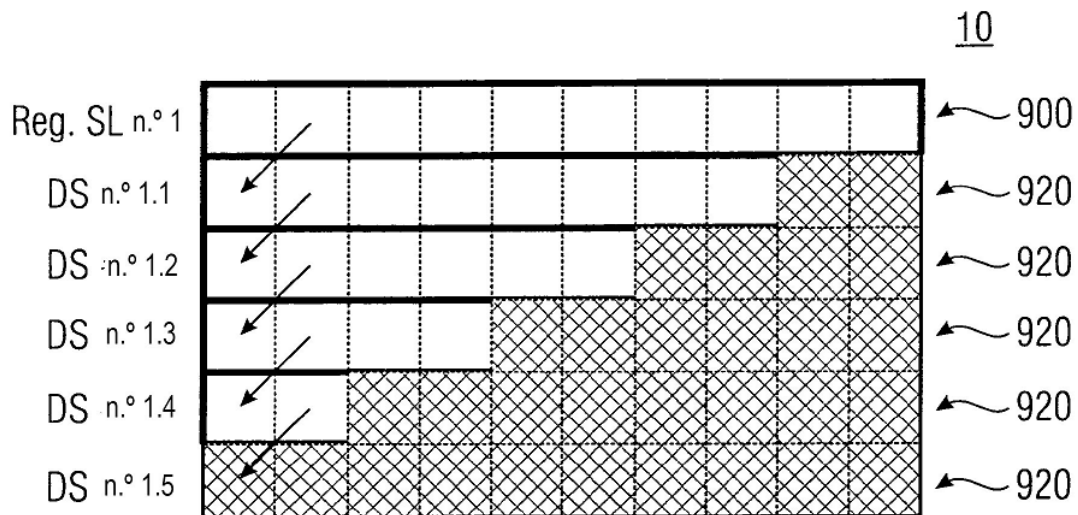


FIG 14

pic_parameter_set_rbsp() {	Descriptor
• • •	
dependent_slices_present_flag	u(1)
last_ctb_cabac_init_flg	u(1)
}	

FIG 15

slice_header() {	Descriptor
first_slice_in_pic_flag	u(1)
si (first_slice_in_pic_flag == 0)	
slice_address	u(v)
slice_type	ue(v)
entropy_slice_flag	u(1)
dependent_slice_flag	u(1)
si (!entropy_slice_flag) {	
si (!dependent_slice_flag) {	
• • •	
} endif {	
no_cabac_reset_flag	u(1)
}	
• • •	
si (cabac_init_present_flag && slice_type != I && !no_cabac_reset_flag)	
cabac_init_flag	u(1)
• • •	
si (slice_type == P slice_type == B)	
five_minus_max_num_merge_cand	ue(v)
si (adaptive_loop_filter_enabled_flag) {	
slice_adaptive_loop_filter_flag	u(1)
si (slice_adaptive_loop_filter_flag && !alf_coef_in_slice_flag)	
alf_param()	
si (slice_adaptive_loop_filter_flag && !alf_coef_in_slice_flag)	
alf_cu_control_param()	
}	
si (seq_loop_filter_across_slices_enabled_flag && (slice_adaptive_loop_filter_flag slice_sample_adaptive_offset_flag !disable_deblocking_filter_flag))	
slice_loop_filter_across_slices_enabled_flag	u(1)
si (tiles_or_entropy_coding_sync_idc > 0) {	
num_entry_point_offsets	ue(v)
si (num_entry_point_offsets > 0) {	
offset_len_minus1	ue(v)
para (i = 0; i < num_entry_point_offsets; i++)	
entry_point_offset[i]	u(v)
}	
}	
}	

FIG 16

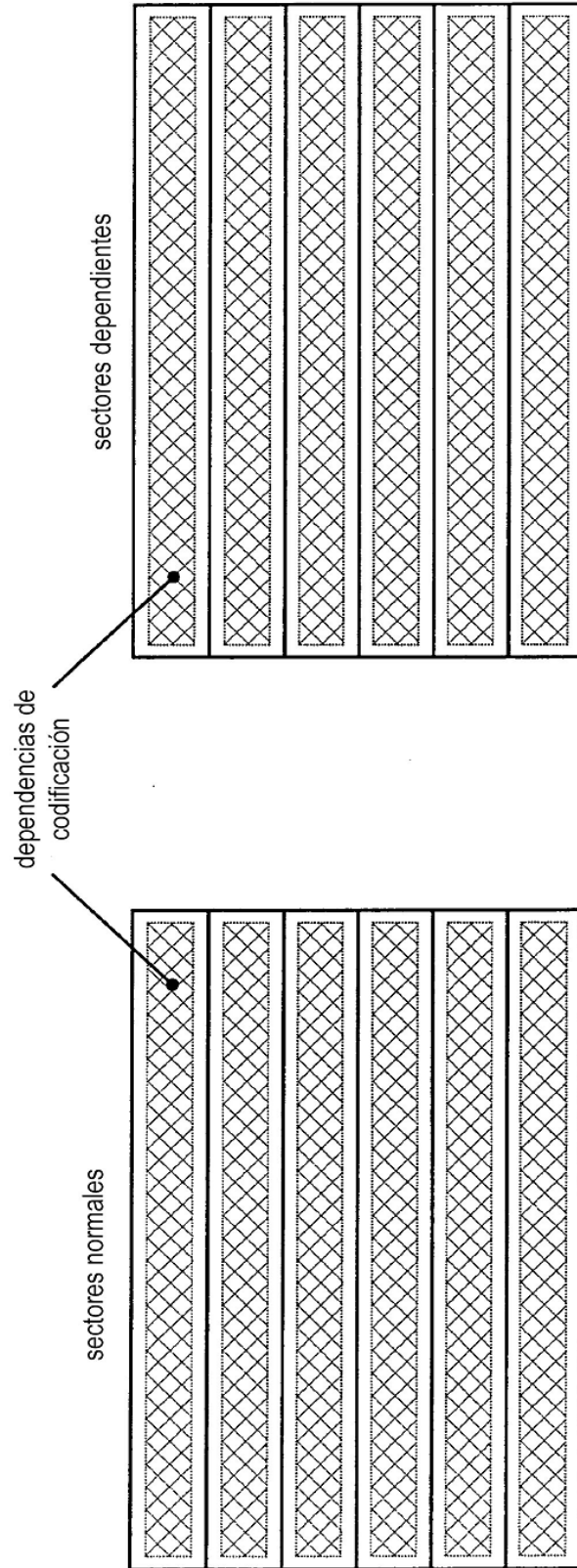


FIG 17

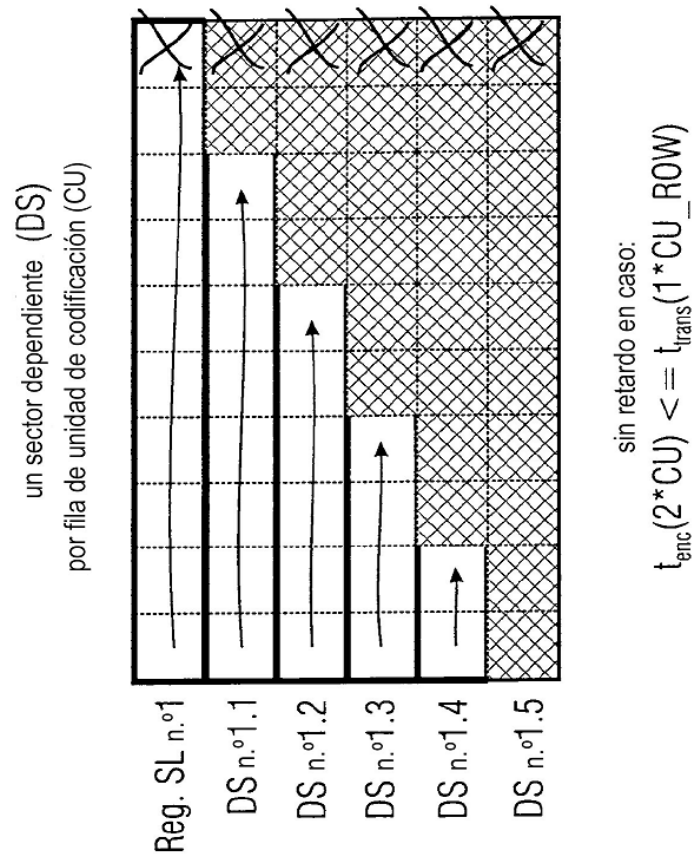


FIG 18

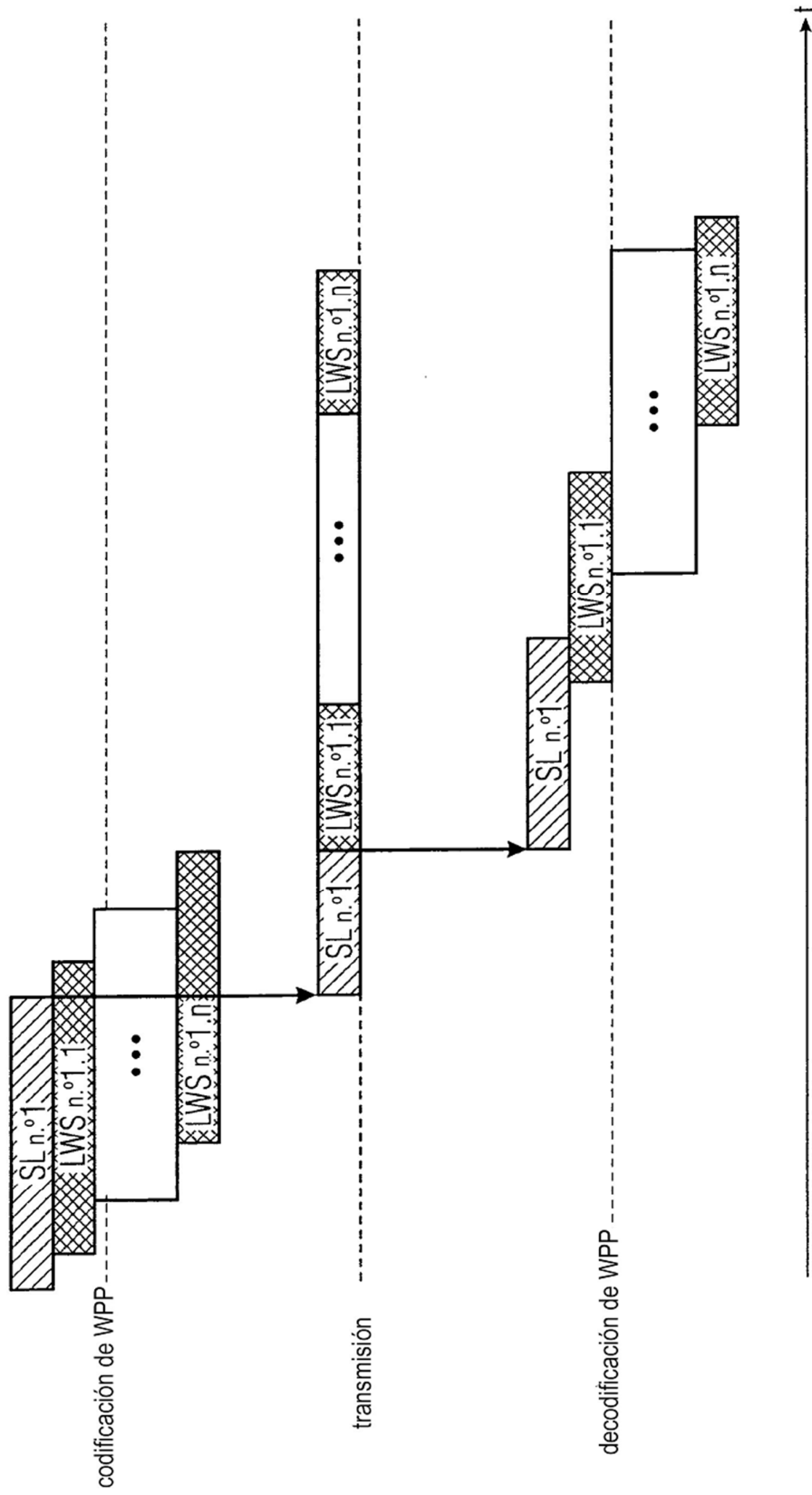


FIG 19

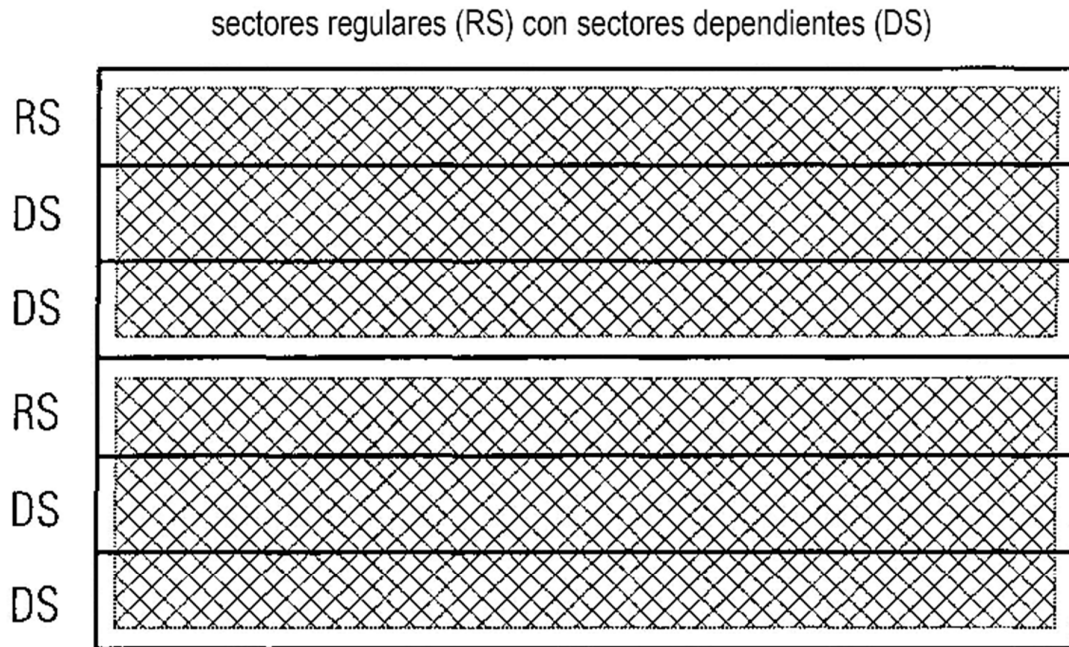


FIG 20

slice_header() {	Descriptor
first_slice_in_pic_flag	u(1)
pic_parameter_set_id	ue(v)
si (!first_slice_in_pic_flag)	
slice_address	u(v)
si (dependent_slice_enabled_flag && !first_slice_in_pic_flag)	
dependent_slice_flag	u(1)
si(!dependent_slice_flag) {	
slice_type	ue(v)
si (output_flag_present_flag)	
pic_output_flag	u(1)
si (separate_colour_plane_flag == 1)	
colour_plane_id	u(2)
si (RapPicFlag) {	
rap_pic_id	ue(v)
no_output_of_prior_pics_flag	u(1)
}	
si (!IdrPicFlag) {	
• • •	
}	
si (tiles_or_entropy_coding_sync_idc == 1 tiles_or_entropy_coding_sync_idc == 2) {	
num_entry_point_offsets	ue(v)
si (num_entry_point_offsets > 0) {	
offset_len_minus1	ue(v)
para (i = 0; i < num_entry_point_offsets; i++)	
entry_point_offset[i]	u(v)
}	
}	
si (slice_header_extension_present_flag) {	
slice_header_extension_length	ue(v)
para (i = 0; i < slice_header_extension_length; i++)	
slice_header_extension_data_byte	u(8)
}	
byte_alignment()	
}	

FIG 21

pic_parameter_set_rbsp() {	Descriptor
pic_parameter_set_id	ue(v)
seq_parameter_set_id	ue(v)
sign_data_hiding_flag	u(1)
si (sign_data_hiding_flag)	
sign_hiding_threshold	u(4)
cabac_init_present_flag	u(1)
num_ref_idx_l0_default_active_minus1	ue(v)
num_ref_idx_l1_default_active_minus1	ue(v)
[Ed. (BB): no presente en software de HM]	
pic_init_qp_minus26	se(v)
constrained_intra_pred_flag	u(1)
slice_granularity	u(2)
diff_cu_qp_delta_depth	ue(v)
cb_qp_offset	se(v)
cr_qp_offset	se(v)
weighted_pred_flag	u(1)
weighted_bipred_idc	u(2)
output_flag_present_flag	u(1)
transquant_bypass_enable_flag	u(1)
dependent_slice_enabled_flag	u(1)
tiles_or_entropy_coding_syne_idc	u(2)
si(tiles_or_entropy_coding_syne_idc $\neq$ 1) {	
num_tile_columns_minus1	ue(v)
num_tile_rows_minus1	ue(v)
uniform_spacing_flag	u(1)
si(!uniform_spacing_flag) {	
para(i = 0; i < num_tile_columns_minus1; i++)	
column_width[i]	ue(v)
para i = 0; i < num_tile_rows_minus1; i++)	
row_height[i]	ue(v)
}	
loop_filter_across_tiles_enabled_flag	u(1)
{ si, no si(tiles_or_entropy_coding_syne_idc $\neq$ 3)	
cabac_independent_flag	u(1)
...	
}	

FIG 22

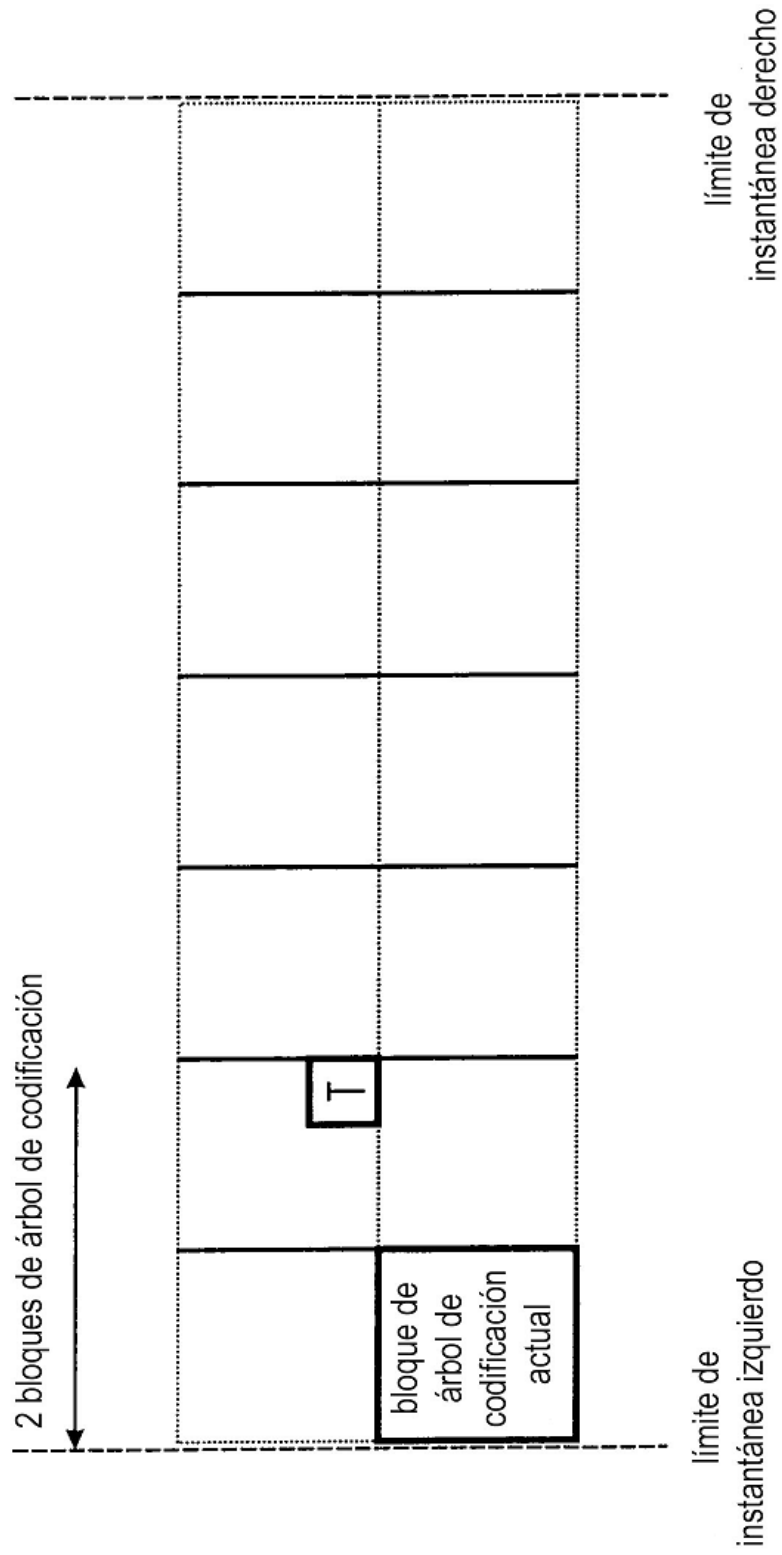
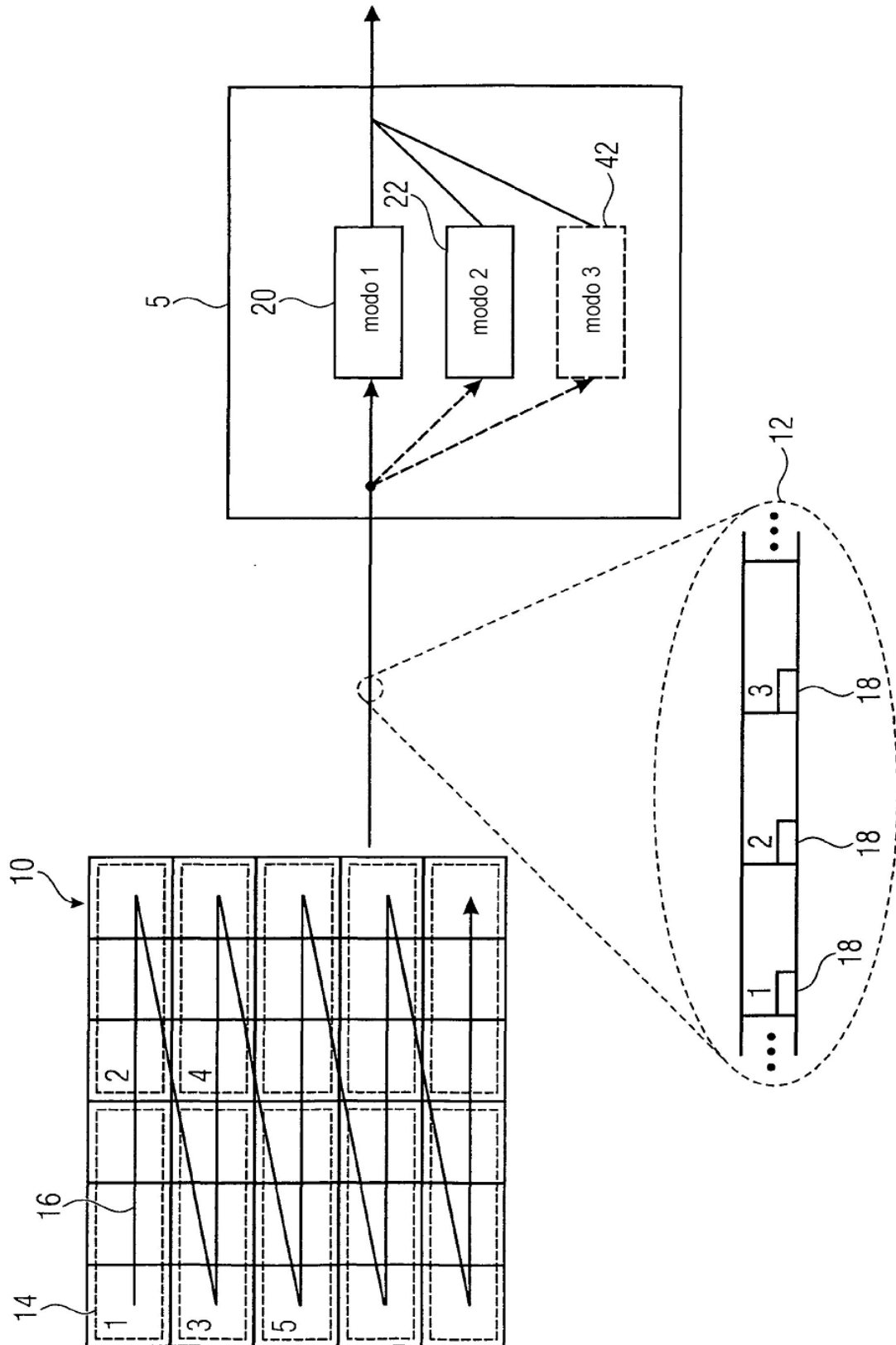


FIG 23



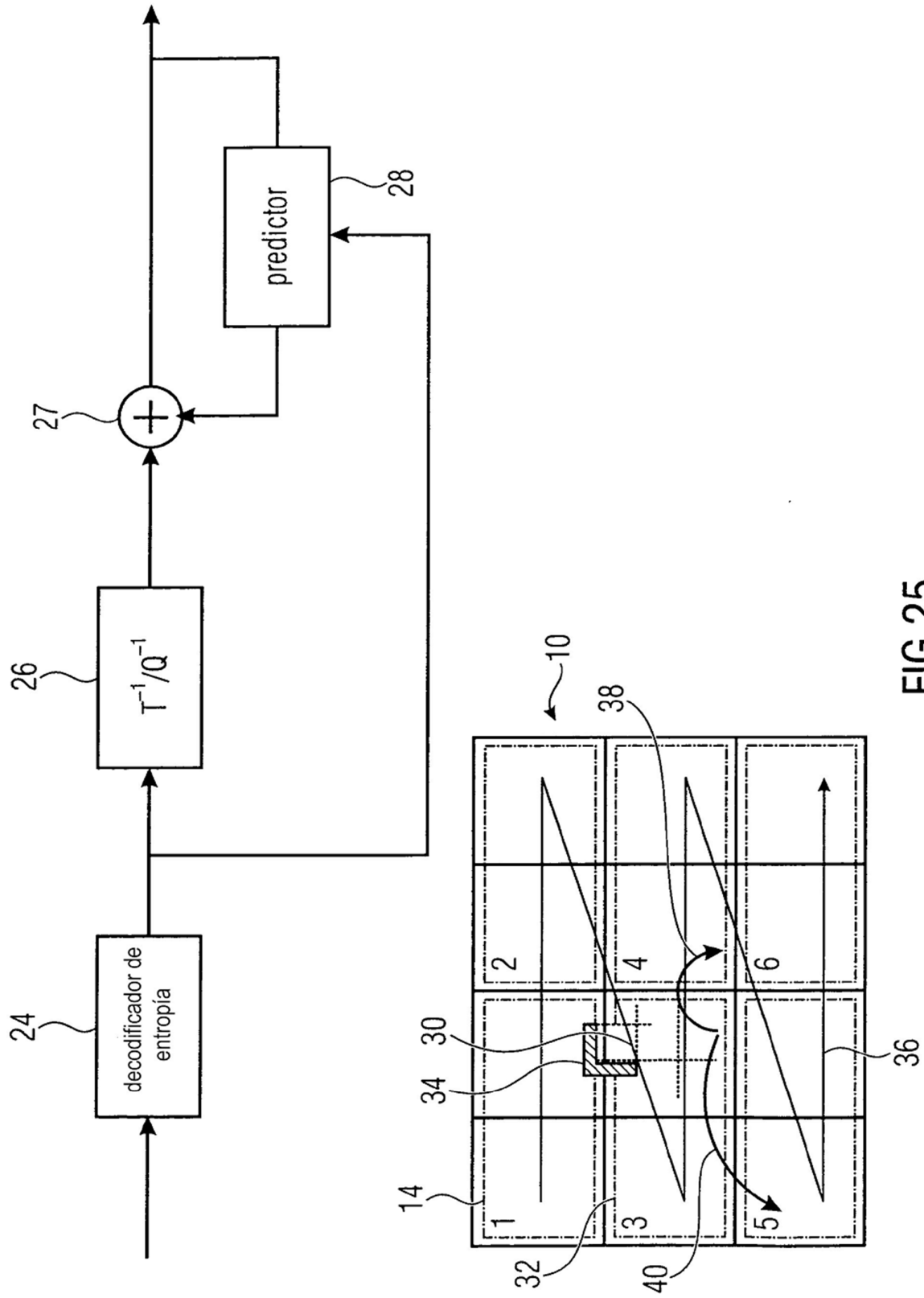


FIG 25

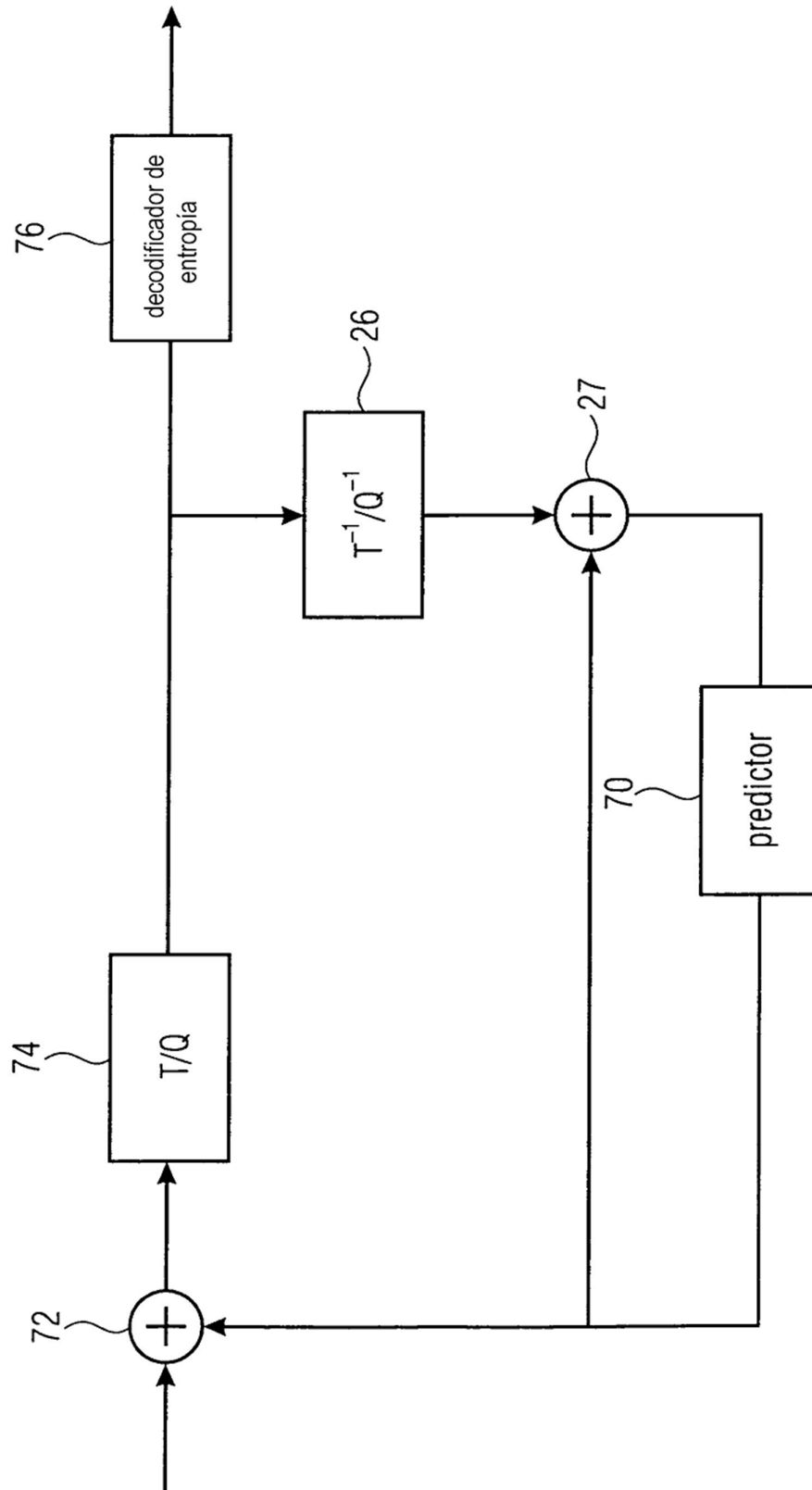


FIG 26

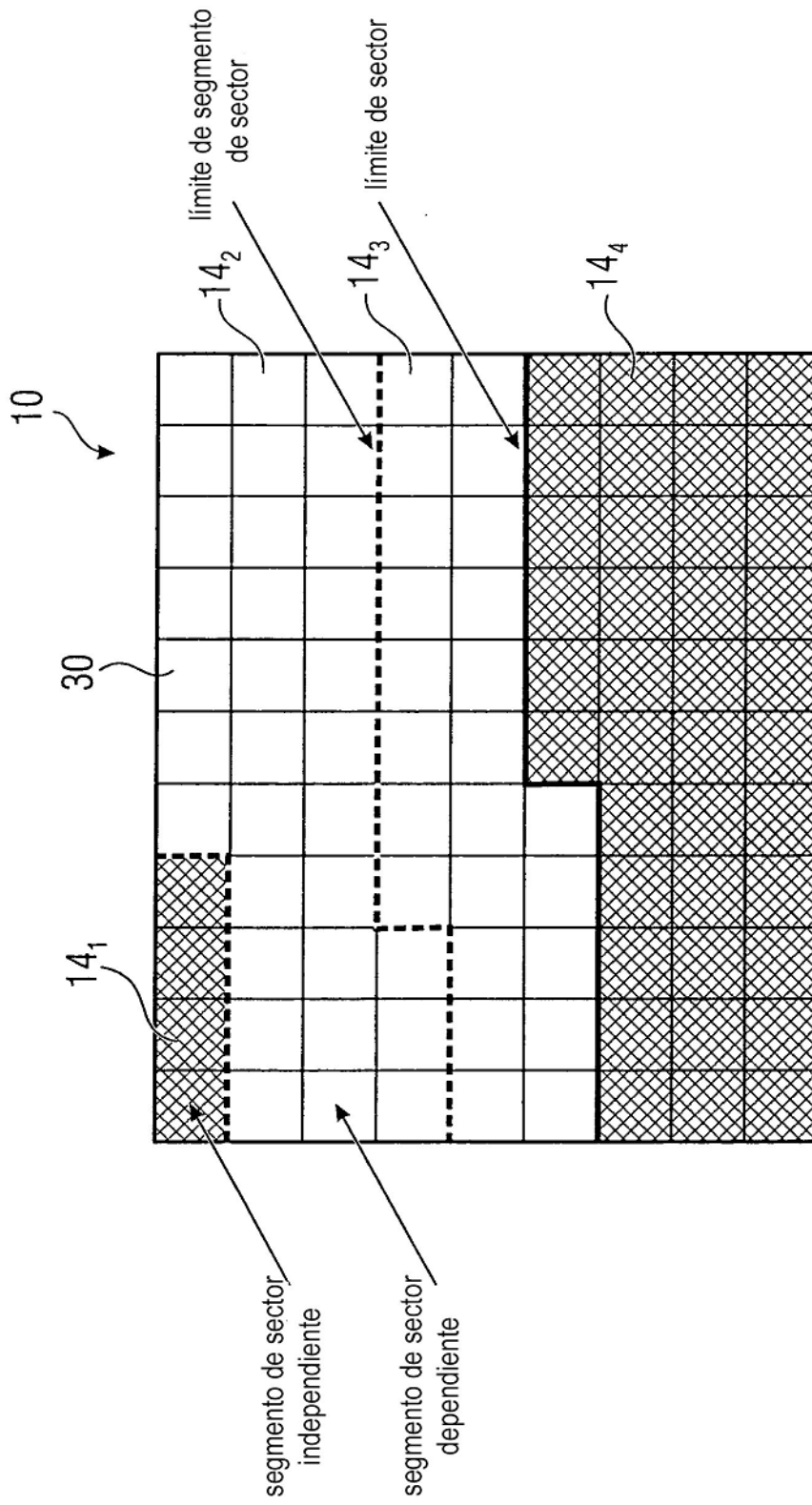


FIG 27

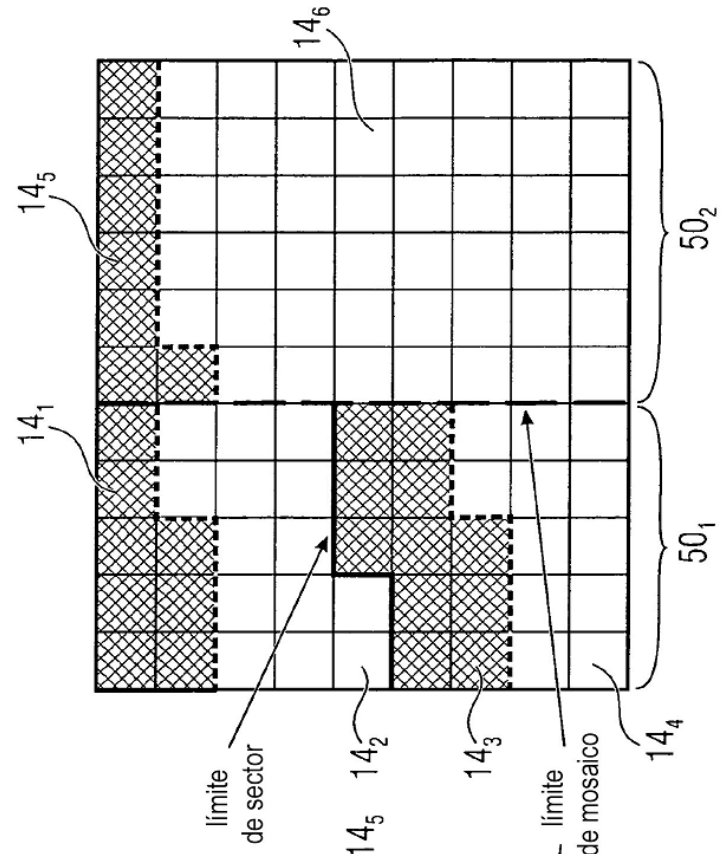
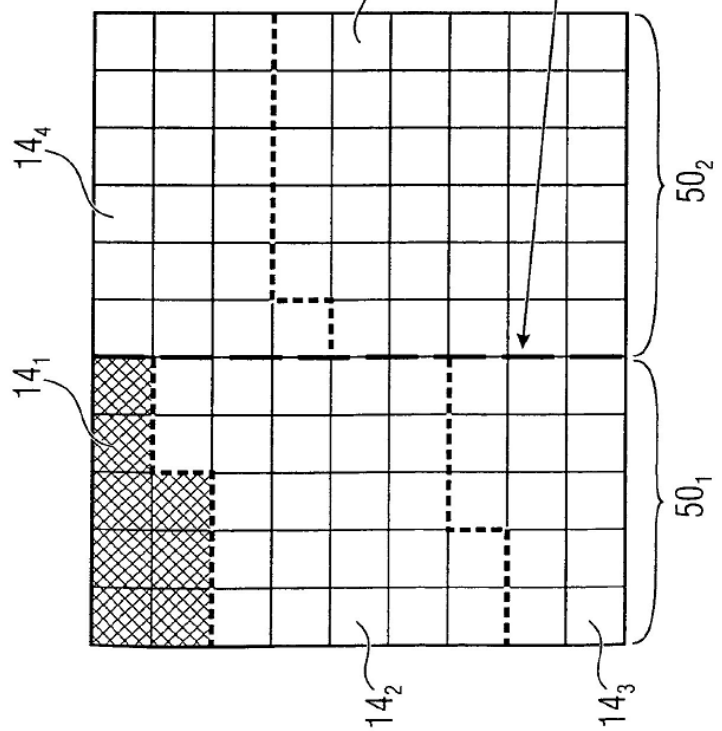


FIG 28B



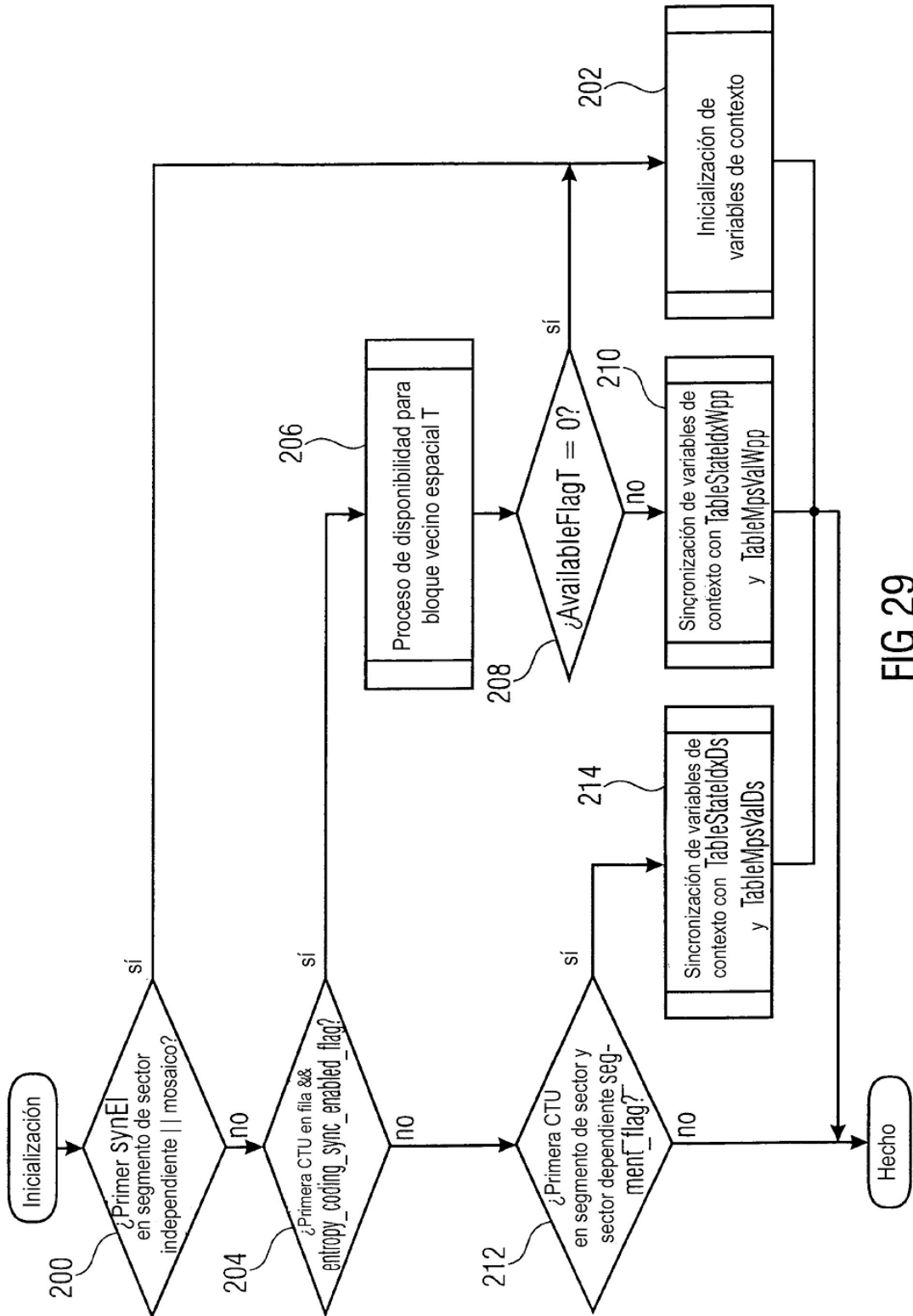


FIG 29

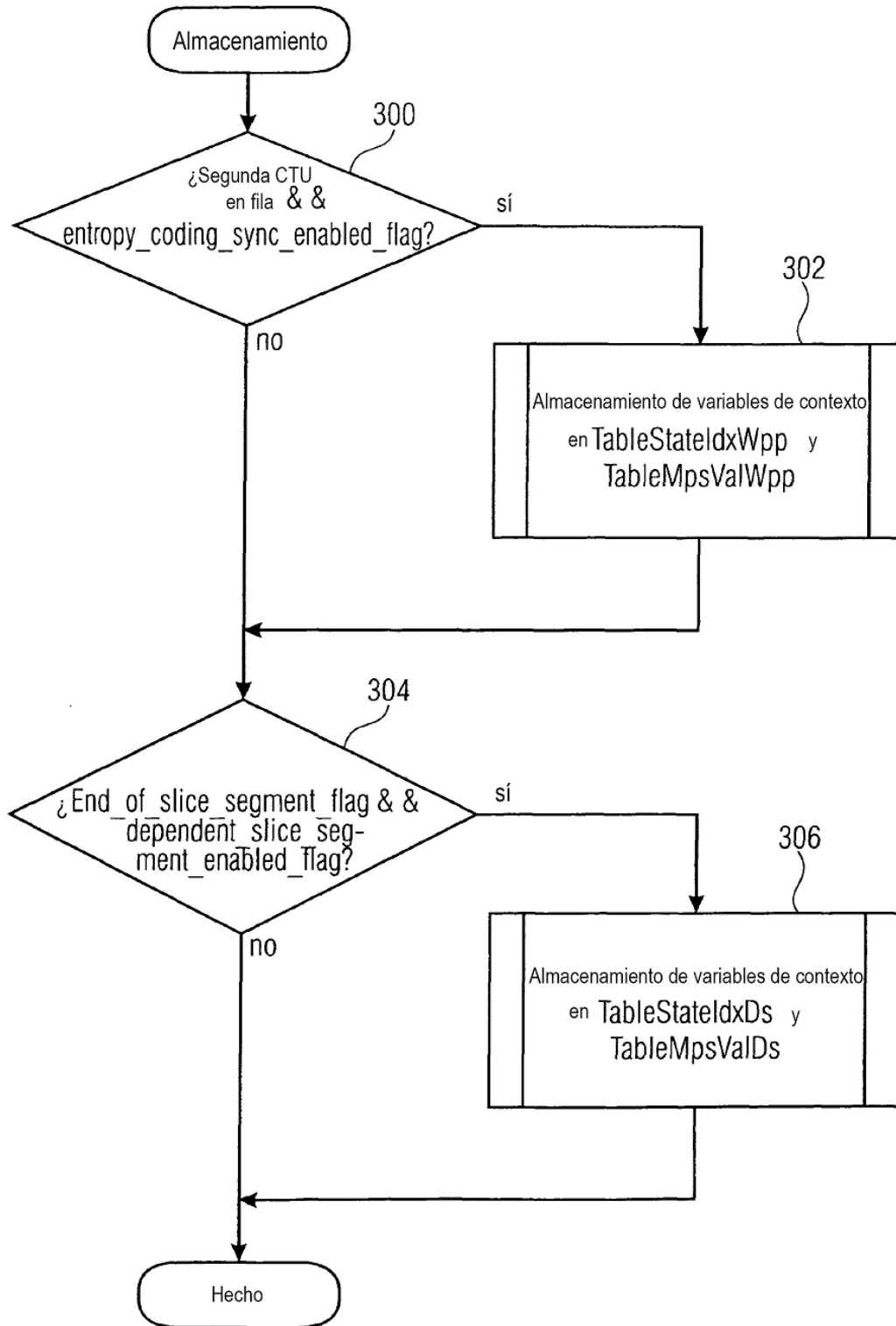


FIG 30

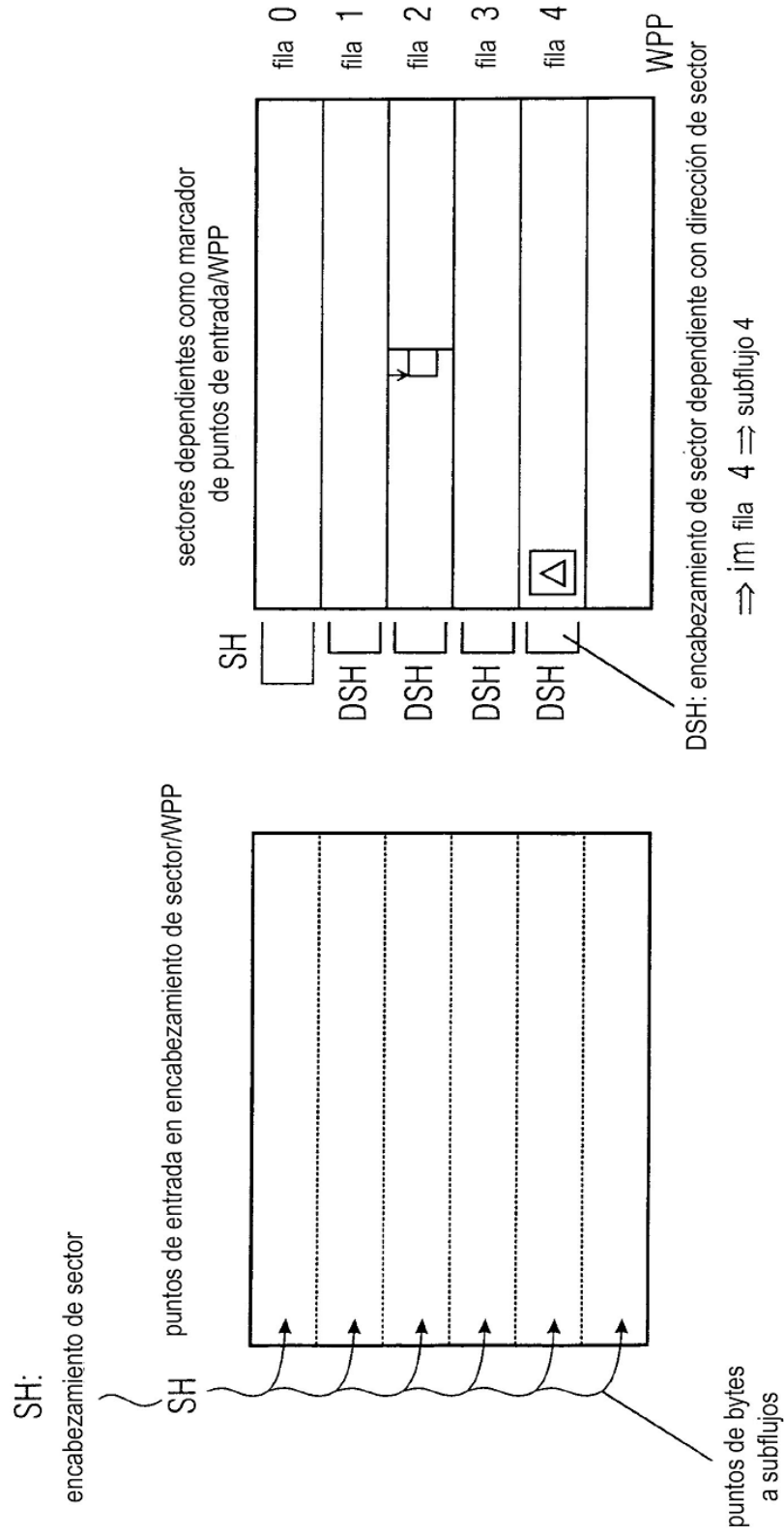


FIG 31