



UNIÓN INTERNACIONAL DE TELECOMUNICACIONES

# UIT-T

SECTOR DE NORMALIZACIÓN  
DE LAS TELECOMUNICACIONES  
DE LA UIT

# H.264

(05/2003)

SERIE H: SISTEMAS AUDIOVISUALES Y  
MULTIMEDIOS

Infraestructura de los servicios audiovisuales –  
Codificación de imágenes vídeo en movimiento

---

**Codificación de vídeo avanzada para los  
servicios audiovisuales genéricos**

Recomendación UIT-T H.264

---

RECOMENDACIONES UIT-T DE LA SERIE H  
SISTEMAS AUDIOVISUALES Y MULTIMEDIOS

CARACTERÍSTICAS DE LOS SISTEMAS VIDEOTELEFÓNICOS	H.100–H.199
INFRAESTRUCTURA DE LOS SERVICIOS AUDIOVISUALES	
Generalidades	H.200–H.219
Multiplexación y sincronización en transmisión	H.220–H.229
Aspectos de los sistemas	H.230–H.239
Procedimientos de comunicación	H.240–H.259
<b>Codificación de imágenes vídeo en movimiento</b>	<b>H.260–H.279</b>
Aspectos relacionados con los sistemas	H.280–H.299
Sistemas y equipos terminales para los servicios audiovisuales	H.300–H.349
Arquitectura de servicios de directorio para servicios audiovisuales y multimedios	H.350–H.359
Arquitectura de la calidad de servicio para servicios audiovisuales y multimedios	H.360–H.369
Servicios suplementarios para multimedios	H.450–H.499
PROCEDIMIENTOS DE MOVILIDAD Y DE COLABORACIÓN	
Visión de conjunto de la movilidad y de la colaboración, definiciones, protocolos y procedimientos	H.500–H.509
Movilidad para los sistemas y servicios multimedios de la serie H	H.510–H.519
Aplicaciones y servicios de colaboración en móviles multimedios	H.520–H.529
Seguridad para los sistemas y servicios móviles multimedios	H.530–H.539
Seguridad para las aplicaciones y los servicios de colaboración en móviles multimedios	H.540–H.549
Procedimientos de interfuncionamiento de la movilidad	H.550–H.559
Procedimientos de interfuncionamiento de colaboración en móviles multimedios	H.560–H.569
SERVICIOS DE BANDA ANCHA Y DE TRÍADA MULTIMEDIOS	
Servicios multimedios de banda ancha sobre VDSL	H.610–H.619

*Para más información, véase la Lista de Recomendaciones del UIT-T.*

## **Recomendación UIT-T H.264**

### **Codificación de vídeo avanzada para los servicios audiovisuales genéricos**

#### **Resumen**

Con la reducción de los precios de la memoria y de la potencia de procesamiento, la diversificación de soporte de vídeo codificado en las redes y los progresos en la tecnología de codificación de vídeo, ha surgido la necesidad de una norma de vídeo comprimido para esta industria con mucha más eficiencia de codificación y una mayor robustez en entornos de red.

Esta Recomendación es una evolución de las normas de codificación de vídeo existentes (H.261, H.262 y H.263) y se elaboró para cubrir la creciente necesidad de mayor compresión de imágenes en movimiento en diversas aplicaciones como videoconferencia, medios de almacenamiento digital, radiodifusión de televisión, emisión por Internet y comunicaciones. Asimismo, se ha concebido para que la representación de vídeo codificado se pueda utilizar de manera flexible en una gran variedad de entornos de red. La utilización de esta Recomendación permite manipular imágenes de vídeo como simples datos digitales, almacenarlas en diversos tipos de medios de almacenamiento, transmitir las y recibirlas por las redes existentes y futuras, y distribuir las por los canales de radiodifusión existentes y futuros.

#### **Orígenes**

La Recomendación UIT-T H.264 fue aprobada el 30 de mayo de 2003 por la Comisión de Estudio 16 (2001-2004) del UIT-T por el procedimiento de la Recomendación UIT-T A.8.

Esta edición incluye las modificaciones introducidas por el corrigendum 1 de la Rec. UIT-T H.264 (2003) aprobado el 7 de mayo de 2004 por la Comisión de Estudio 16 del UIT-T según el procedimiento de la Recomendación UIT-T A.8.

## PREFACIO

La UIT (Unión Internacional de Telecomunicaciones) es el organismo especializado de las Naciones Unidas en el campo de las telecomunicaciones. El UIT-T (Sector de Normalización de las Telecomunicaciones de la UIT) es un órgano permanente de la UIT. Este órgano estudia los aspectos técnicos, de explotación y tarifarios y publica Recomendaciones sobre los mismos, con miras a la normalización de las telecomunicaciones en el plano mundial.

La Asamblea Mundial de Normalización de las Telecomunicaciones (AMNT), que se celebra cada cuatro años, establece los temas que han de estudiar las Comisiones de Estudio del UIT-T, que a su vez producen Recomendaciones sobre dichos temas.

La aprobación de Recomendaciones por los Miembros del UIT-T es el objeto del procedimiento establecido en la Resolución 1 de la AMNT.

En ciertos sectores de la tecnología de la información que corresponden a la esfera de competencia del UIT-T, se preparan las normas necesarias en colaboración con la ISO y la CEI.

## NOTA

En esta Recomendación, la expresión "Administración" se utiliza para designar, en forma abreviada, tanto una administración de telecomunicaciones como una empresa de explotación reconocida de telecomunicaciones.

La observancia de esta Recomendación es voluntaria. Ahora bien, la Recomendación puede contener ciertas disposiciones obligatorias (para asegurar, por ejemplo, la aplicabilidad o la interoperabilidad), por lo que la observancia se consigue con el cumplimiento exacto y puntual de todas las disposiciones obligatorias. La obligatoriedad de un elemento preceptivo o requisito se expresa mediante las frases "tener que, haber de, hay que + infinitivo" o el verbo principal en tiempo futuro simple de mandato, en modo afirmativo o negativo. El hecho de que se utilice esta formulación no entraña que la observancia se imponga a ninguna de las partes.

## PROPIEDAD INTELECTUAL

La UIT señala a la atención la posibilidad de que la utilización o aplicación de la presente Recomendación suponga el empleo de un derecho de propiedad intelectual reivindicado. La UIT no adopta ninguna posición en cuanto a la demostración, validez o aplicabilidad de los derechos de propiedad intelectual reivindicados, ya sea por los miembros de la UIT o por terceros ajenos al proceso de elaboración de Recomendaciones.

En la fecha de aprobación de la presente Recomendación, la UIT ha recibido notificación de propiedad intelectual, protegida por patente, que puede ser necesaria para aplicar esta Recomendación. Sin embargo, debe señalarse a los usuarios que puede que esta información no se encuentre totalmente actualizada al respecto, por lo que se les insta encarecidamente a consultar la base de datos sobre patentes de la TSB.

© UIT 2005

Reservados todos los derechos. Ninguna parte de esta publicación puede reproducirse por ningún procedimiento sin previa autorización escrita por parte de la UIT.

# ÍNDICE

	<b>Página</b>
Prefacio .....	x
0 Introducción.....	xi
0.1 Prólogo .....	xi
0.2 Objetivo .....	xi
0.3 Aplicaciones .....	xi
0.4 Perfiles y niveles.....	xii
0.5 Panorama general de las características de diseño .....	xii
0.6 Instrucciones para leer esta especificación.....	xiv
1 Alcance .....	1
2 Referencias normativas.....	1
3 Definiciones.....	1
4 Abreviaturas, siglas o acrónimos .....	12
5 Convenios .....	13
5.1 Operadores aritméticos .....	13
5.2 Operadores lógicos .....	13
5.3 Operadores relacionales.....	13
5.4 Operadores de bit.....	14
5.5 Operadores de asignación.....	14
5.6 Notación para indicar una gama de valores.....	14
5.7 Funciones matemáticas.....	14
5.8 Variables, elementos sintácticos y cuadros .....	15
5.9 Descripción mediante texto de operaciones lógicas .....	16
5.10 Procesos.....	17
6 Formato de los datos fuente, codificados, decodificados y de salida, proceso de barrido y relaciones de adyacencia .....	17
6.1 Formato del tren de bits .....	17
6.2 Formato de imágenes fuente, decodificadas y resultantes.....	18
6.3 Subdivisión espacial de imágenes y sectores .....	20
6.4 Procesos de barrido inverso y procesos de obtención de macrobloques adyacentes.....	21
7 Sintaxis y semántica .....	33
7.1 Métodos de describir la sintaxis en forma tabular.....	33
7.2 Especificación de la sintaxis de funciones, categorías y descriptores.....	35
7.3 Sintaxis en forma tabular.....	37
7.4 Semántica .....	51
8 Proceso de decodificación .....	95
8.1 Proceso de decodificación de unidades NAL.....	96
8.2 Proceso de decodificación de sectores .....	97

	<b>Página</b>
8.3	Proceso de predicción intra ..... 121
8.4	Proceso de predicción inter ..... 134
8.5	Proceso de decodificación de coeficientes de transformada y proceso de reconstrucción de imágenes antes de aplicar el proceso de filtrado de bloques..... 160
8.6	Proceso de decodificación de macrobloques P en sectores SP o macrobloques SI ..... 168
8.7	Proceso de filtrado de bloques..... 174
9	Proceso de análisis sintáctico ..... 185
9.1	Proceso de análisis sintáctico de códigos Exp-Golomb ..... 185
9.2	Proceso de análisis CAVLC de niveles de coeficientes de transformada ..... 189
9.3	Proceso de análisis CABAC de datos de sector ..... 197
Anexo A	– Perfiles y niveles..... 240
A.1	Requisitos relativos a la capacidad del decodificador de vídeo ..... 240
A.2	Perfiles..... 241
A.3	Niveles..... 242
Anexo B	– Formato del tren de bytes ..... 251
B.1	Sintaxis y semántica de unidades NAL de tren de bytes..... 251
B.2	Proceso de decodificación de unidades NAL de tren de bytes..... 252
B.3	Recuperación de la alineación por byte del decodificador (informativo) ..... 253
Anexo C	– Decodificador ficticio de referencia ..... 254
C.1	Funcionamiento de la memoria intermedia de imágenes codificadas (CPB). 256
C.2	Funcionamiento de la memoria intermedia de imágenes decodificadas (DPB)..... 259
C.3	Conformidad del tren de bits ..... 261
C.4	Conformidad del decodificador ..... 262
Anexo D	– Información de perfeccionamiento complementaria ..... 267
D.1	Sintaxis de la cabida útil SEI..... 268
D.2	Semántica de la cabida útil del mensaje SEI ..... 276
Anexo E	– Información sobre los posibles usos de vídeo..... 299
E.1	Sintaxis de la información sobre los posibles usos de vídeo (VUI) ..... 300
E.2	Semántica de la información sobre uso de vídeo (VUI)..... 301

## LISTA DE FIGURAS

	<b>Página</b>
Figura 6-1 – Posiciones vertical y horizontal nominales de muestras luma y croma 4:2:0 en un cuadro.....	19
Figura 6-2 – Posiciones de muestreo vertical y horizontal nominales de los campos de muestra superior e inferior.....	20
Figura 6-3 – Subdivisión de una imagen de 11 por 9 macrobloques en dos sectores.....	20
Figura 6-4 – Subdivisión de un cuadro decodificado en pares de macrobloques.....	21
Figura 6-5 – Particiones macrobloque, particiones submacrobloque, barrido de particiones macrobloque y barrido de particiones submacrobloque.....	22
Figure 6-6 – Barrido de bloques luma 4x4.....	23
Figura 6-7 – Macrobloques adyacentes a un determinado macrobloque.....	24
Figura 6-8 – Macrobloques adyacentes de un determinado macrobloque en cuadros MBAFF.....	25
Figura 6-9 – Determinación de macrobloques, bloques y particiones adyacentes (informativo).....	26
Figura 7-1 – Estructura de una unidad de acceso que no contiene unidades NAL cuyos nal_unit_type sean 0, 7, 8 o estén entre 12 y 31, inclusive.....	59
Figura 8-1 – Direcciones del modo de predicción Intra_4x4 (informativo).....	123
Figura 8-2 – Obtención de vector de movimiento en modo directo temporal (informativo)...	146
Figura 8-3 – Predicción de segmentación direccional (informativo).....	147
Figura 8-4 – Posiciones de muestras enteras (bloques sombreados con letras mayúsculas) y posiciones de muestras fraccionarias (bloques sin sombreado y en letras minúsculas) para la interpolación luma de un cuarto de muestra.....	153
Figura 8-5 – Variables dependientes de posición de muestra fraccionaria en la interpolación croma y las muestras de posición entera de alrededor A, B, C y D.....	155
Figura 8-6 – Asignación de los índices de dcY a luma4x4BlkIdx.....	161
Figure 8-7 – Asignación de los índices dcC a chroma4x4BlkIdx.....	162
Figura 8-8 – a) Barrido en Zig-zag. b) Barrido por campo.....	163
Figura 8-9 – Bordos de un macrobloque que se va a filtrar (los bordes luma se muestran con línea continua y los croma con líneas discontinuas).....	174
Figura 8-10 – Convenio para describir muestras que están en el contorno horizontal o vertical de un bloque 4x4.....	178
Figura 9-1 – Ilustración del proceso de análisis CABAC de un elemento sintáctico SE (informativo).....	199
Figura 9-2 – Descripción del proceso de decodificación aritmética de un solo bin (informativo).....	228
Figura 9-3 – Diagrama de flujo de la decodificación de una decisión.....	230
Figura 9-4 – Diagrama de flujo de la renormalización.....	232
Figura 9-5 – Diagrama de flujo del proceso de codificación de derivación.....	233

Figura 9-6 – Diagrama de flujo de la decodificación de una decisión antes de la terminación .....	234
Figura 9-7 – Diagrama de flujo de la codificación de una decisión .....	236
Figura 9-8 – Diagrama de flujo de la renormalización en el codificador .....	237
Figura 9-9 – Diagrama de flujo de PutBit(B) .....	237
Figura 9-10 – Diagrama de flujo de la codificación de derivación.....	238
Figura 9-11 – Diagrama de flujo de la codificación de una decisión antes de la terminación	239
Figura 9-12 – Diagrama de flujo de la evacuación en la terminación .....	239
Figura C-1 – Estructura de los trenes de bytes y de los trenes de unidades NAL para verificar la conformidad mediante el HRD .....	254
Figura C-2 – Modelo de memoria intermedia HRD .....	255
Figura E-1 – Ubicación de las muestras cromas de los campos superior e inferior en función de chroma_sample_loc_type_top_field y chroma_sample_loc_type_bottom_field ....	307



## LISTA DE CUADROS

	<b>Página</b>
Cuadro 6-1 – Valores de ChromaFormatFactor.....	18
Cuadro 6-2 – Especificación de las asignaciones de argumento y resultado que se utilizan en las subcláusulas 6.4.7.1 a 6.4.7.5 .....	26
Cuadro 6-3 – Especificación de mbAddrN .....	30
Cuadro 6-4 – Especificación de mbAddrN e yM.....	32
Cuadro 7-1 – Códigos de tipos de unidades NAL .....	53
Cuadro 7-2 – Significado de primary_pic_type.....	67
Cuadro 7-3 – Nombres correspondientes a slice_type.....	70
Cuadro 7-4 – Operaciones reordering_of_pic_nums_idc para reordenar las listas de imágenes de referencia .....	77
Cuadro 7-5 – Interpretación de adaptive_ref_pic_marking_mode_flag.....	80
Cuadro 7-6 – Valores de operación de control de gestión de memoria (memory_management_control_operation) .....	81
Cuadro 7-7 – Tipos de macrobloques genéricos posibles en función de slice_type.....	84
Cuadro 7-8 – Tipos de macrobloques de sectores I .....	85
Cuadro 7-9 – Tipo de macrobloque con valor 0 correspondiente a sectores SI.....	86
Cuadro 7-10 – Tipos de macrobloques de valores entre 0 a 4 correspondiente a sectores P y SP.....	87
Cuadro 7-11 – Tipo de macrobloque de valores 0 a 22 correspondiente a sectores B .....	88
Cuadro 7-12 – Especificación de los valores de CodedBlockPatternChroma .....	90
Cuadro 7-13 – Relación entre los modos de predicción Intra_chroma_pred_mode y espacial .....	90
Cuadro 7-14 – Tipos de submacrobloque en macrobloques P .....	91
Cuadro 7-15 – Tipos de submacrobloque en macrobloques B .....	92
Cuadro 8-1 – Tipo de mapeado de grupos de sectores refinado .....	103
Cuadro 8-2 – Especificación de Intra4x4PredMode[ luma4x4BlkIdx ] y sus correspondientes nombres asociados .....	123
Cuadro 8-3 – Especificación de Intra16x16PredMode y sus correspondientes nombres.....	129
Cuadro 8-4 – Especificación de los modos de predicción intra croma y sus correspondientes nombres .....	131
Cuadro 8-5 – Especificación de la variable colPic .....	139
Cuadro 8-6 – Especificación de PicCodingStruct( X ) .....	139
Cuadro 8-7 – Especificación de mbAddrCol, yM, y vertMvScale .....	140
Cuadro 8-8 – Asignación de los indicadores de utilización de predicción .....	143
Cuadro 8-9 – Cálculo de la componente vertical del vector croma en modo de codificación campo.....	149
Cuadro 8-10 – Posiciones luma de muestra entera diferenciales.....	153

Cuadro 8-11 – Asignación de la muestra de predicción luma $\text{predPartLXL}[x_L, y_L]$ .....	155
Cuadro 8-12 – Especificación del mapeado de $\text{idx}$ en $c_{ij}$ para el barrido en zig-zag y por campo.....	163
Cuadro 8-13 – Especificación de $\text{QP}_C$ como una función de $\text{qP}_I$ .....	164
Cuadro 8-14 – Cálculo de $\text{indexA}$ e $\text{indexB}$ a partir de las variables umbral $\alpha$ y $\beta$ dependientes de la traslación .....	183
Cuadro 8-14 (continuación)– Cálculo de $\text{indexA}$ e $\text{indexB}$ a partir de las variables umbral $\alpha$ y $\beta$ dependientes de la traslación .....	183
Cuadro 8-15 – Valor de la variable de recorte del filtro $t_{C0}$ en función de $\text{indexA}$ y $bS$ .....	184
Cuadro 8-15 (continuación) – Valor de la variable de recorte del filtro $t_{C0}$ en función de $\text{indexA}$ y $bS$ .....	184
Cuadro 9-1 – Cadena de bits con bits "prefijo" y "sufijo" y asignación de la gama de valores de $\text{codeNum}$ (informativo).....	186
Cuadro 9-2 – Cadenas de bits Exp-Golomb y $\text{codeNum}$ en forma explícita y utilizadas como $\text{ue}(v)$ (informativo) .....	186
Cuadro 9-3 – Asignación a $\text{codeNum}$ del elemento sintáctico para elementos sintácticos codificados mediante Exp-Golomb con signo $\text{se}(v)$ .....	187
Cuadro 9-4 – Asignación a $\text{codeNum}$ de valores del $\text{coded\_block\_pattern}$ para los modos de predicción de macrobloque .....	188
Cuadro 9-5 – Mapeado de $\text{coeff\_token}$ en $\text{TotalCoeff}(\text{coeff\_token})$ y $\text{TrailingOnes}(\text{coeff\_token})$ .....	192
Cuadro 9-6 – Palabras de código para $\text{level\_prefix}$ .....	195
Cuadro 9-7 – $\text{total\_zeros}$ correspondiente a bloques 4x4 para valores de $\text{TotalCoeff}(\text{coeff\_token})$ entre 1 y 7 .....	196
Cuadro 9-8 – $\text{total\_zeros}$ correspondiente a bloques 4x4 para valores de $\text{TotalCoeff}(\text{coeff\_token})$ entre 8 y 15 .....	196
Cuadro 9-9 – $\text{total\_zeros}$ correspondiente a bloques 2x2 DC croma.....	197
Cuadro 9-10 – $\text{run\_before}$ en función de $\text{zerosLeft}$ .....	197
Cuadro 9-11 – Asociación de $\text{ctxIdx}$ y elementos sintácticos para cada tipo de sector en el proceso de inicialización.....	201
Cuadro 9-12 – Valores de las variables $m$ y $n$ para $\text{ctxIdx}$ entre 0 y 10 .....	201
Cuadro 9-13 – Valores de las variables $m$ y $n$ para $\text{ctxIdx}$ entre 11 y 23 .....	202
Cuadro 9-14 – Valores de las variables $m$ y $n$ para $\text{ctxIdx}$ entre 24 y 39 .....	202
Cuadro 9-15 – Valores de las variables $m$ y $n$ para $\text{ctxIdx}$ entre 40 y 53 .....	202
Cuadro 9-16 – Valores de las variables $m$ y $n$ para $\text{ctxIdx}$ entre 54 y 59 .....	202
Cuadro 9-17 – Valores de las variables $m$ y $n$ para $\text{ctxIdx}$ entre 60 y 69 .....	203
Cuadro 9-18 – Valores de las variables $m$ y $n$ para $\text{ctxIdx}$ entre 70 y 104 .....	203
Cuadro 9-19 – Valores de las variables $m$ y $n$ para $\text{ctxIdx}$ entre 105 y 165 .....	204
Cuadro 9-20 – Valores de las variables $m$ y $n$ para $\text{ctxIdx}$ entre 166 y 226 .....	205

	<b>Página</b>
Cuadro 9-21 – Valores de las variables m y n para ctxIdx entre 227 y 275 .....	206
Cuadro 9-22 – Valores de las variables m y n para ctxIdx entre 277 y 337 .....	207
Cuadro 9-23 – Valores de las variables m y n para ctxIdx entre 338 y 398 .....	208
Cuadro 9-24 – Elementos sintácticos y tipos de binarización asociados, maxBinIdxCtx y ctxIdxOffset .....	210
Cuadro 9-25 – Cadena bin de la binarización unaria (informativo) .....	211
Cuadro 9-26 – Binarización para tipo de macrobloque en sectores SI .....	214
Cuadro 9-27 – Binarización de tipo de macrobloque en sectores P, SP y B .....	215
Cuadro 9-28 – Binarización para tipos de submacrobloques en sectores P, SP y B.....	216
Cuadro 9-29 – Asignación de ctxIdxInc a binIdx para todos los valores de ctxIdxOffset excepto aquéllos relacionados con los elementos sintácticos coded_block_flag, significant_coeff_flag, last_significant_coeff_flag, y oeff_abs_level_minus1 .....	218
Cuadro 9-30 – Asignación de ctxIdxBlockCatOffset a ctxBlockCat para los elementos sintácticos coded_block_flag, significant_coeff_flag, last_significant_coeff_flag, y coeff_abs_level_minus1 .....	219
Cuadro 9-31 – Especificación de ctxIdxInc en función de los valores de ctxIdxOffset y binIdx.....	226
Cuadro 9-32 – Especificación de ctxBlockCat para los diferentes bloques .....	227
Cuadro 9-33 – Especificación de rangeTabLPS en función de pStateIdx y de qCodIRangeIdx.....	231
Cuadro 9-34 – Tabla de transiciones de estado.....	232
Cuadro A-1 – Límites de nivel.....	244
Cuadro A-2 – Límites de nivel del perfil básico .....	246
Cuadro A-3 – Límites de nivel del perfil principal.....	246
Cuadro A-4 – Límites de nivel del perfil extendido .....	247
Cuadro A-5 – Velocidades máximas de cuadro (cuadros por segundo) para algunos ejemplos de tamaños de cuadros.....	248
Cuadro D-1 – Interpretación de pic_struct .....	278
Cuadro D-2 – Correspondencia de ct_type con el barrido de la imagen fuente .....	279
Cuadro D-3 – Definición de los valores counting_type.....	279
Cuadro D-4 – Valores de scene_transition_type.....	289
Cuadro E-1 – Significado del indicador de relación de aspecto de las muestras.....	302
Cuadro E-2 – Significado de video_format .....	303
Cuadro E-3 – Colores primarios .....	304
Cuadro E-4 – Características de transferencia .....	305
Cuadro E-5 – Coeficientes de la matriz .....	306
Cuadro E-6 – Divisor para el cálculo de $\Delta t_{fi,dpb}(n)$ .....	308

## **Prefacio**

La UIT (Unión Internacional de Telecomunicaciones) es el organismo especializado de las Naciones Unidas en el campo de las telecomunicaciones. El UIT-T (Sector de Normalización de las Telecomunicaciones de la UIT) es un órgano permanente de la UIT. Este órgano estudia los aspectos técnicos, de explotación y tarifarios y publica Recomendaciones sobre los mismos, con miras a la normalización de las telecomunicaciones en el plano mundial. La Asamblea Mundial de Normalización de las Telecomunicaciones (AMNT), que se celebra cada cuatro años, establece los temas que han de estudiar las Comisiones de Estudio del UIT-T, que a su vez producen Recomendaciones sobre dichos temas. La aprobación de Recomendaciones UIT-T es el objeto del procedimiento establecido en la Resolución 1 de la AMNT. En ciertos sectores de la tecnología de la información que corresponden a la esfera de competencia del UIT-T, se preparan las normas necesarias en colaboración con la ISO y la CEI.

La ISO (International Organization for Standardisation) y la CEI (Comisión Electrotécnica Internacional) forman el sistema especializado para la normalización en el plano mundial. Los organismos nacionales que son miembros de la ISO y de la CEI participan en la elaboración de Normas Internacionales a través de Comités Técnicos establecidos por la respectiva organización para tratar determinados campos de actividad técnica. Los Comités Técnicos de la ISO y de la CEI colaboran en campos de interés mutuo. También participan en el trabajo otras organizaciones internacionales, gubernamentales y no gubernamentales, en coordinación con la ISO y la CEI. En el campo de la tecnología de la información, la ISO y la CEI han creado un Comité Técnico Mixto, el JTC 1 de ISO/CEI. Los proyectos de Normas Internacionales adoptados por el Comité Técnico Mixto se someten a la votación de los organismos nacionales. Para la publicación como una Norma Internacional se requiere la aprobación, por lo menos, del 75% de los organismos nacionales que votan.

Esta Recomendación | Norma Internacional fue el resultado de una colaboración entre la CE 16 C.6 del UIT-T, también conocida como VCEG (Grupo de expertos en codificación de vídeo) y el JTC1/SC29/WG 11 de la ISO/CEI, también conocido como MPEG (Grupo de expertos en imágenes en movimiento). El VCEG se formó en 1997 para mantener actualizadas las normas anteriores de codificación de vídeo del UIT-T y elaborar nuevas normas de codificación de vídeo adecuadas para una gran variedad de servicios conversacionales y no conversacionales. El MPEG se estableció en 1988 con el fin de elaborar normas de codificación de imágenes en movimiento y su correspondiente audio para diversas aplicaciones, por ejemplo medios de almacenamiento digital, distribución, y comunicación.

En esta Recomendación | Norma Internacional los anexos A a E contienen requisitos normativos y son parte integral de esta Recomendación | Norma Internacional.

## 0 Introducción

Esta cláusula no es parte integrante de esta Recomendación | Norma Internacional.

### 0.1 Prólogo

Esta subcláusula no es parte integrante de esta Recomendación | Norma Internacional.

Con la reducción de los precios de la memoria y de la potencia de procesamiento, la diversificación de soporte de datos de vídeo codificado en las redes y los progresos en la tecnología de codificación de vídeo, ha surgido la necesidad de una norma de representación de vídeo comprimido para esta industria con mucha más eficiencia de codificación y una mayor robustez en entornos de red. A tal efecto, el Grupo de expertos en codificación de vídeo (VCEG, *video coding experts group*) del UIT-T y el Grupo de expertos en imágenes en movimiento (MPEG, *moving picture experts group*) de la ISO/CEI formaron el Grupo mixto de vídeo (JVT, *joint video team*) en 2001 para elaborar una nueva Recomendación | Norma Internacional.

### 0.2 Objetivo

Esta subcláusula no es parte integrante de esta Recomendación | Norma Internacional.

Esta Recomendación | Norma Internacional se elaboró para cubrir la creciente necesidad de mayor compresión de imágenes en movimiento en diversas aplicaciones como videoconferencia, medios de almacenamiento digital, radiodifusión de televisión, emisión por Internet y comunicaciones. Asimismo, se ha concebido para que la representación de vídeo codificado se pueda utilizar de manera flexible en una gran variedad de entornos de red. La utilización de esta Recomendación | Norma Internacional permite manipular imágenes de vídeo como simples datos digitales, almacenarlas en diversos tipos de medios de almacenamiento, transmitir las y recibir las por las redes existentes y futuras y distribuir las por los canales de radiodifusión existentes y futuros.

### 0.3 Aplicaciones

Esta subcláusula no es parte integrante de esta Recomendación | Norma Internacional.

Esta Recomendación | Norma Internacional está concebida para abarcar una extensa gama de aplicaciones de vídeo, entre las que se cuentan:

- CATV Televisión por cable en redes ópticas, de cobre, etc. (*cable TV on optical networks, copper, etc.*)
- DBS Servicios de vídeo de radiodifusión directa por satélite (*direct broadcast satellite video services*)
- DSL Servicios de vídeo por línea de abonado digital (*digital subscriber line video services*)
- DTTB Radiodifusión de televisión terrenal digital (*digital terrestrial television broadcasting*)
- ISM Medios de almacenamiento interactivos (discos ópticos, etc.) (*interactive storage media (optical disks, etc.)*)
- MMM Distribución de correo multimedios (*multimedia mailing*)
- MSPN Servicios multimedios por redes de paquetes (*multimedia services over packet networks*)
- RTC Servicios conversacionales en tiempo real (videoconferencia, videoteléfono, etc.) (*real-time conversational services (videoconferencing, videophone, etc.)*)
- RVS Vigilancia por vídeo a distancia (*remote video surveillance*)

#### **0.4 Perfiles y niveles**

Esta subcláusula no es parte integrante de esta Recomendación | Norma Internacional.

Esta Recomendación | Norma Internacional está concebida para ser genérica en el sentido de que sirva para una gran variedad de aplicaciones, velocidades de bits, resoluciones, niveles de calidad y servicios. Deben quedar abarcadas aplicaciones tales como los medios de almacenamiento digitales, la radiodifusión de televisión y las comunicaciones en tiempo real. Al elaborar esta especificación se tuvieron en cuenta diversos requisitos de las aplicaciones típicas, se elaboraron los algoritmos necesarios y se integró todo esto en una única sintaxis. Así pues, esta especificación facilita el intercambio de datos de vídeo entre diferentes aplicaciones.

Teniendo en cuenta las consideraciones prácticas en cuanto a la implementación de toda la sintaxis de esta especificación, se ha creado una serie de subconjuntos de la sintaxis mediante la definición de "perfiles" y "niveles". Éstos y otros términos se definen formalmente en la cláusula 3.

Un "perfil" es un subconjunto de toda la sintaxis de tren de bits que se especifica en esta Recomendación | Norma Internacional. Aun con las restricciones impuestas a la sintaxis de un determinado perfil es posible exigir una gran variedad en la calidad de funcionamiento de los codificadores y decodificadores en función de los valores que tomen los elementos sintácticos del tren de bits, por ejemplo el tamaño especificado de las imágenes decodificadas. En muchas aplicaciones, no resulta práctico ni económico implementar un decodificador capaz de tratar con todos los posibles usos de la sintaxis correspondiente a un determinado perfil.

Para solucionar este problema, se especifican "niveles" en cada perfil. Un nivel es un conjunto concreto de restricciones de los valores de los elementos sintácticos del tren de bits. Estas restricciones pueden consistir simplemente en límites de los valores. También es posible que se trate de limitaciones sobre combinaciones aritméticas de valores (por ejemplo, la anchura de la imagen multiplicado por la altura de la imagen multiplicado por el número de imágenes decodificadas por segundo).

El contenido vídeo codificado que cumple esta Recomendación | Norma Internacional utiliza una sintaxis común. Para lograr un subconjunto de la sintaxis completa se incluyen en el tren de bits indicadores, parámetros y otros elementos sintácticos que indican la presencia de elementos sintácticos que aparecen posteriormente en el tren de bits.

#### **0.5 Panorama general de las características de diseño**

Esta subcláusula no es parte integrante de esta Recomendación | Norma Internacional.

La representación codificada especificada en la sintaxis está concebida para obtener una gran capacidad de compresión con la calidad de imagen deseada. No se trata de un algoritmo sin pérdidas, pues normalmente los valores exactos de las muestras originales no se preservan después de aplicar los procesos de codificación y decodificación. Es posible utilizar varias técnicas para lograr una compresión muy eficiente. Los algoritmos de codificación (que no se especifican en esta Recomendación | Norma Internacional) pueden utilizar codificación inter o codificación intra para regiones en forma de bloque de cada imagen. La codificación inter utiliza vectores en movimiento para la predicción inter de bloques con objeto de explotar la dependencia temporal estadística entre imágenes diferentes. La codificación intra utiliza diversos modos de predicción espacial para explotar las dependencias espaciales estadísticas en la señal original de una misma imagen. Los vectores en movimiento y los modos de predicción intra se pueden especificar para diversos tamaños de bloque de la imagen. El residuo de la predicción se comprime nuevamente utilizando una transformada para eliminar la correlación espacial dentro del bloque de transformada antes de que sea cuantificado; se trata de un proceso irreversible que normalmente descarta la información

visual menos importante y genera una buena aproximación de las muestras originales. Por último, los vectores en movimiento o los modos de predicción intra se pueden combinar con la información de los coeficientes de transformada cuantificados y codificar utilizando códigos de longitud variable o la codificación aritmética.

### **0.5.1 Codificación predictiva**

Esta subcláusula no es parte integrante de esta Recomendación | Norma Internacional.

Debido al dilema entre las necesidades de acceso aleatorio y de una compresión muy eficiente, se especifican dos tipos principales de codificación. La codificación intra no utiliza referencias a otras imágenes. Si bien la codificación intra puede proporcionar puntos de acceso a la secuencia codificada a partir de los cuales se puede empezar o continuar correctamente la decodificación, normalmente sólo se puede lograr una eficiencia de compresión moderada. La codificación inter (predictiva o bipredictiva) es más eficiente y utiliza predicción inter de cada bloque de valores de muestra correspondientes a imágenes decodificadas anteriormente, y que fueron seleccionadas por el codificador. A diferencia de otras normas de codificación de vídeo, las imágenes codificadas utilizando predicción inter bipredictiva también se pueden utilizar como referencias para la codificación inter de otras imágenes.

La aplicación de estos tres tipos de codificación a una secuencia de imágenes es flexible y, por lo general, el orden de decodificación no es el mismo que el orden de adquisición de la imagen original en el codificador o el orden de salida del decodificador para la visualización. La elección entre un tipo u otro se deja al codificador y depende de los requisitos de la aplicación. El orden de decodificación se especifica de manera que la decodificación de imágenes que utiliza predicción inter se aplica después, en orden de decodificación, que las imágenes a las que se hace referencia en el proceso de decodificación.

### **0.5.2 Codificación de vídeo progresivo y entrelazado**

Esta subcláusula no es parte integrante de esta Recomendación | Norma Internacional.

Esta Recomendación | Norma Internacional especifica la sintaxis y el proceso de decodificación de vídeo que fue creado mediante barrido progresivo o barrido entrelazado, los cuales pueden aparecer juntos en la misma secuencia. El tiempo de adquisición es distinto para los dos campos de un cuadro entrelazado, y es el mismo para los dos campos de un cuadro progresivo. Se puede codificar cada campo por separado o codificar los dos campos juntos como si se tratara de un cuadro. Normalmente los cuadros progresivos se codifican como un solo cuadro. En el caso de vídeo entrelazado, el codificador puede elegir entre la codificación cuadro o la codificación campo, lo cual se puede seleccionar adaptativamente imagen por imagen y también de manera más localizada dentro de un cuadro codificado. Normalmente se prefiere la codificación cuadro cuando la escena vídeo contiene un grado importante de detalle y no hay mucho movimiento. La codificación campo funciona mejor cuando hay un movimiento rápido entre imágenes.

### **0.5.3 Subdivisión de la imagen en macrobloques y particiones más pequeñas**

Esta subcláusula no es parte integrante de esta Recomendación | Norma Internacional.

Al igual que en las anteriores Recomendaciones y Normas Internacionales sobre codificación de vídeo, un macrobloque consiste en un bloque de 16x16 de muestras luma y sus dos correspondientes bloques de muestras croma, y se utiliza como la unidad de procesamiento básica del proceso de decodificación de vídeo.

Los macrobloques se pueden subdividir aún más para realizar la predicción inter. El tamaño de las particiones de predicción inter es el resultado de un compromiso entre la ganancia de codificación que se consigue mediante la compensación de movimiento con bloques más pequeños y el volumen de datos necesarios para presentar los datos de una compensación de movimiento. En esta Recomendación | Norma Internacional el tamaño más pequeño que se utiliza para representar

movimiento en la predicción inter es de 4x4 muestras luma, para lo cual se utiliza una resolución del vector en movimiento de un cuarto de la distancia de separación de la cuadrícula de muestras luma. Para la predicción inter de un bloque de muestras también se puede seleccionar la imagen que se utilizará como imagen de referencia a partir de una serie de imágenes almacenadas decodificadas previamente. Los vectores de movimiento se codifican de manera diferente que los valores predichos formados a partir de los vectores de movimiento codificados cercanos.

Normalmente el codificador calcula los vectores de movimiento adecuados y otros elementos de datos representados en el tren de datos de vídeo. Este proceso de estimación del movimiento en el codificador y la selección de si se utiliza predicción inter para representar cada región del contenido de vídeo no forma parte de los objetivos de esta Recomendación | Norma Internacional.

#### **0.5.4 Reducción de la redundancia espacial**

Esta subcláusula no es parte integrante de esta Recomendación | Norma Internacional.

Las imágenes originales y el residuo de la predicción contienen mucha redundancia espacial. Esta Recomendación | Norma Internacional se basa en la utilización del método de la transformada por bloque para eliminar la redundancia espacial. Después de realizar la predicción inter a partir de las muestras de otras imágenes decodificadas anteriormente o predicciones espaciales a partir de muestras decodificadas previamente de la imagen actual, el residuo de predicción resultante se divide en bloques de 4x4. Una vez cuantificados, estos bloques se convierten al dominio de la transformada. Después de la cuantificación muchos de los coeficientes de transformada son cero o tienen una amplitud pequeña, lo que permite representarlos con muy pocos datos codificados. En esta Recomendación | Norma Internacional no se especifican los procesos de transformación y de cuantificación.

#### **0.6 Instrucciones para leer esta especificación**

Esta cláusula no es parte integrante de esta Recomendación | Norma Internacional.

Se sugiere que el lector empiece por la cláusula 1 (Alcance) y siga por la cláusula 3 (Definiciones). En la cláusula 6 se describe la relación geométrica de la imagen original, la entrada y la salida del decodificador. En la cláusula 7 (Sintaxis y semántica) se especifica el orden de análisis de los elementos sintácticos del tren de bits. En las subcláusulas 7.1-7.3 se describe el orden sintáctico y en la subcláusula 7.4 la semántica; es decir, el alcance, las restricciones y las condiciones que se imponen a los elementos sintácticos. El análisis sintáctico real para la mayoría de los elementos sintácticos se especifica en la cláusula 9 (Proceso de análisis sintáctico). Por último, en la cláusula 8 (Proceso de decodificación) se especifica cómo se obtienen las muestras decodificadas a partir de los elementos sintácticos. Al leer esta especificación el lector debe referirse, cuando lo considere necesario, a las cláusulas 2 (Referencias normativas), 4 (abreviaturas, siglas o acrónimos) y 5 (Convenios). Los anexos A a E también son parte integrante de esta Recomendación | Norma Internacional.

En el anexo A se definen tres perfiles (básico, principal y extendido), siendo cada uno de ellos adecuado para ciertos dominios de aplicación, y se definen también los denominados niveles de los perfiles. En el anexo B se especifica la sintaxis y la semántica de un formato de tren de bytes para el suministro de vídeo codificado en la forma de un tren de bytes ordenado. En el anexo C se describe un decodificador ficticio de referencia y su utilización para comprobar la conformidad del tren de bits y del decodificador. En el anexo D se especifica la sintaxis y la semántica para cabidas útiles de mensajes de información de perfeccionamiento complementaria. Por último el anexo E especifica la sintaxis y la semántica de los parámetros de información relativa a los posibles usos de vídeo del conjunto de parámetros secuencia.

En esta especificación, los párrafos que empiezan con "NOTA –" son informativos y no son parte integrante de esta Recomendación | Norma Internacional.



## Recomendación UIT-T H.264

### Codificación de vídeo avanzada para los servicios audiovisuales genéricos

#### 1 Alcance

Este documento contiene la Recomendación UIT-T H.264 | Norma Internacional ISO/CEI 14496-10 sobre codificación de vídeo.

#### 2 Referencias normativas

Las siguientes Recomendaciones y Normas Internacionales contienen disposiciones que, mediante su referencia en este texto, constituyen disposiciones de esta Recomendación | Norma Internacional. A la fecha de esta publicación, las ediciones citadas están en vigor. Todas las Recomendaciones y Normas son objeto de revisión, por lo que se alienta a los usuarios de esta Recomendación | Norma Internacional a que utilicen la edición más reciente de las Recomendaciones y Normas que se indican a continuación. Los miembros de la CEI y de la ISO mantienen un registro actualizado de las Normas Internacionales vigentes. La Oficina de Normalización de las Telecomunicaciones publica periódicamente una lista de las Recomendaciones UIT-T vigentes.

- Recomendación UIT-T T.35 (2000), *Procedimiento para la asignación de códigos definidos por el UIT-T para facilidades no normalizadas*.
- ISO/CEI 11578:1996, Annex A, *Universal Unique Identifier*.
- ISO/CEI 10527:1991, *Colorimetric Observers*.

#### 3 Definiciones

A los efectos de la presente Recomendación | Norma Internacional, se aplican las siguientes definiciones:

- 3.1 unidad de acceso:** Conjunto de *unidades NAL* que contienen siempre exactamente una *imagen codificada primaria*. Además de la *imagen codificada primaria*, la unidad de acceso también puede contener *imágenes codificadas redundantes* u otras *unidades NAL* que no contienen *sectores* o *particiones de datos de sectores* de una *imagen codificada*. La decodificación de una unidad de acceso siempre resulta en una *imagen decodificada*.
- 3.2 coeficiente de transformada c.a.:** *Coefficientes de transformada* cuyo *índice de frecuencia* en una o en dos dimensiones es distinto de cero.
- 3.3 proceso de decodificación aritmética binaria:** *Proceso de decodificación* entrópica que deriva los valores de los *bin* a partir de un *tren de bits* generado por un *proceso de codificación aritmética binaria adaptativa*.
- 3.4 proceso de codificación aritmética binaria adaptativa:** *Proceso de codificación* entrópica, cuya norma no se especifica en esta Recomendación | Norma Internacional, que codifica una secuencia de *bins* y genera un *tren de bits* que se puede decodificar mediante el *proceso de decodificación aritmética binaria adaptativa*.
- 3.5 orden de sectores arbitrario:** *Orden de decodificación* de *sectores* en el que la *dirección del macrobloque* del primer *macrobloque* de algún *sector* de la *imagen* puede ser más pequeña que la *dirección del macrobloque* del primer *macrobloque* de algún otro *sector* anterior de la misma *imagen codificada*.
- 3.6 sector B:** *Sector* que se puede decodificar mediante una *predicción intra* a partir de las muestras decodificadas del mismo *sector* o mediante una *predicción inter* de *imágenes de referencia* previamente decodificadas, utilizando a lo sumo dos *vectores de movimiento* e *índices de referencia* para *predecir* los valores de las muestras de cada *bloque*.

- 3.7 bin:** Un bit de una *cadena bin*.
- 3.8 binarización:** Conjunto de *cadena bin* para todos los posibles valores de un *elemento sintáctico*.
- 3.9 proceso de binarización:** Proceso de mapeado unívoco de todos los valores posibles de un *elemento sintáctico* en un conjunto de *cadena bin*.
- 3.10 cadena bin:** Cadena de *bins*. Una cadena bin es una representación binaria intermedia de los valores de los *elementos sintácticos* que surge durante la *binarización* de un *elemento sintáctico*.
- 3.11 sector bipredictivo:** Véase *sector B*.
- 3.12 tren de bits:** Secuencia de bits que constituye la representación de *imágenes codificadas* y sus correspondientes datos, y que forma una o más *secuencias de vídeo codificado*. "Tren de bits" es un nombre genérico que designa un *tren de unidades NAL* o un *tren de bytes*.
- 3.13 bloque:** Matriz MxN (M-columnas por N-filas) de muestras, o una matriz MxN de *coeficientes de transformada*.
- 3.14 campo inferior:** Uno de los dos *campos* que constituyen un *cuadro*. Cada fila de un campo *inferior* está ubicada justo debajo de la fila correspondiente del campo *superior*.
- 3.15 macrobloque inferior (de un par de macrobloques):** *Macrobloque* dentro de un *par de macrobloques* que contiene las muestras en la fila inferior de las muestras del *par de macrobloques*. En un *par de macrobloques campo*, el macrobloque inferior representa las muestras de la región del campo *inferior* del *cuadro* que está en la región espacial del *par de macrobloques*. En un *par de macrobloques cuadro*, el macrobloque inferior representa las muestras del *cuadro* que están en la mitad inferior de la región espacial del *par de macrobloques*.
- 3.16 enlace interrumpido:** Posición en el *tren de bits* en la que se indica que algunas de las *imágenes* subsiguientes por *orden de decodificación* pueden estar distorsionadas debido a operaciones no especificadas realizadas al generar el *tren de bits*.
- 3.17 byte u octeto:** Secuencia de 8 bits, escritos o leídos, con el bit más significativo a la izquierda y el menos significativo a la derecha. Cuando se trata de una secuencia de bits de datos, el primer bit del byte es el más significativo.
- 3.18 alineado por byte:** Se dice que una posición de un *tren de bits* está alineada por byte si la posición es un entero múltiplo de 8 bits a partir del primer bit del *tren de bits*. Se dice que un bit o *byte* o *elemento sintáctico* está alineado por *byte* cuando la posición en la que aparece en un *tren de bits* está alineada por byte.
- 3.19 tren de bytes:** Encapsulado de un *tren de unidades NAL* que contiene *prefijos de código de inicio* y *unidades NAL* como se especifica en el anexo B.
- 3.19.1 poder:** Verbo utilizado para referirse a un comportamiento que aunque es permitido no es necesariamente exigido.
- 3.20 categoría:** Número asociado a cada *elemento sintáctico*. La categoría se utiliza para especificar la atribución de *elementos sintácticos* a *unidades NAL* para *subdividir en particiones datos de sector*. También se puede utilizar, de un modo que depende de la aplicación, para hacer referencia a clases de *elementos sintácticos*, lo cual no se especifica en esta Recomendación | Norma Internacional.
- 3.21 croma:** Adjetivo que califica a una muestra o matriz de muestras que representa una de las dos señales de diferencia de color relativas a los colores primarios. Los símbolos de una matriz o muestra croma son Cb y Cr.

NOTA – Se utiliza el término "croma" en lugar de "crominancia" porque éste último se asocia con la utilización de características de transferencia de luz lineal.

- 3.22 campo codificado:** *Representación codificada de un campo.*
- 3.23 cuadro codificado:** *Representación codificada de un cuadro.*
- 3.24 imagen codificada:** *Representación codificada de una imagen.* Puede tratarse de un *campo codificado* o de un *cuadro codificado*. "Imagen codificada" es un nombre genérico que designa una *imagen codificada primaria* o una *imagen codificada redundante*, pero no las dos.
- 3.25 memoria intermedia de imágenes codificadas (CPB, coded picture buffer):** Memoria intermedia FIFO (primero en entrar, primero en salir) que contiene *unidades de acceso* en el *orden de decodificación* especificado en el *decodificador de referencia ficticio* del anexo C.
- 3.26 representación codificada:** Elemento de datos en su forma codificada.
- 3.27 secuencia de vídeo codificado:** Secuencia de *unidades de acceso* que consta de, por orden de decodificación, una *unidad de acceso IDR* seguida por ninguna o varias *unidades de acceso distintas de las IDR*, incluidas todas las subsiguientes *unidades de acceso* hasta la siguiente *unidad de acceso IDR*, sin contar esta última.
- 3.28 componente:** Muestra o matriz de una de las tres matrices (una *luma* y dos *croma*) que constituyen un *campo* o un *cuadro*.
- 3.29 par de campos complementarios:** Nombre genérico que designa un *par de campos complementarios de referencia* o un *par de campos complementarios distintas de las de referencia*.
- 3.30 par de campos complementarios distintos de los de referencia:** Dos *campos distintos de los de referencia* que están en *unidades de acceso* consecutivas en *orden de decodificación* como *campos codificados* de paridad contraria, en el que el primer *campo* todavía no es un *campo apareado*.
- 3.31 par de campos complementarios de referencia:** Dos *campos de referencia* que están en *unidades de acceso* consecutivas por *orden de decodificación*, como *campos codificados* y tienen el mismo *valor del elemento sintáctico* `frame_num`, donde el segundo *campo* por *orden de decodificación* no es una *imagen IDR* y no incluye un *elemento sintáctico* `memory_management_control_operation` (control de gestión de memoria) de valor igual a 5.
- 3.32 variable contexto:** Variable de un *proceso de decodificación aritmética binaria adaptativa* de un *bin*, especificada mediante una ecuación que contiene los *bins* decodificados recientemente.
- 3.33 coeficiente de transformada c.c.:** *Coeficiente de transformada* cuyo *índice de frecuencia* es cero en todas las dimensiones.
- 3.34 imagen decodificada:** Imagen que se obtiene al decodificar una *imagen codificada*. Una *imagen decodificada* es un *cuadro* decodificado o un *campo* decodificado. Este último es un *campo superior* decodificado o un *campo inferior* decodificado.
- 3.35 memoria intermedia de imágenes decodificadas (DPB, decoded picture buffer):** Memoria intermedia que almacena las *imágenes decodificadas* para tenerlas como referencia, reordenar la salida, o retrasar la salida según especifica el *decodificador de referencia ficticio* en el anexo C.
- 3.36 decodificador:** Dispositivo que ejecuta el *proceso de decodificación*.

- 3.37 orden de decodificación:** Orden en el que el *proceso de decodificación* procesa los *elementos sintácticos*.
- 3.38 proceso de decodificación:** Proceso especificado en esta Recomendación | Norma Internacional, que a partir del *tren de bits* que lee, deduce *imágenes decodificadas*.
- 3.39 predicción directa:** *Predicción inter* aplicable a un *bloque* que no tiene decodificado ningún *vector de movimiento*. Se especifican dos modos de *predicción*, a saber, modo de *predicción* directa espacial y modo de *predicción* temporal.
- 3.40 decodificador en prueba (DUT, decoder under test):** *Decodificador* para el que se comprueba la conformidad con esta Recomendación | Norma Internacional, para lo cual se aplica el *planificador de tren ficticio* para suministrar al *decodificador* y al *decodificador de referencia ficticio* un *tren de bits* conforme, y luego se comparan los valores y tiempos del resultado de los dos *decodificadores*.
- 3.41 byte de prevención de emulación:** Byte de valor 0x03 que puede aparecer en una *unidad NAL*. La presencia de bytes de prevención de emulación garantizan que ninguna secuencia de bytes consecutivos alienados por byte en la *unidad NAL* contiene un *prefijo de código de inicio*.
- 3.42 codificador:** Dispositivo que ejecuta el *proceso de codificación*.
- 3.43 proceso de codificación:** Proceso, no especificado en esta Recomendación | Norma Internacional, que genera un *tren de bits* conforme con esta Recomendación | Norma Internacional.
- 3.44 campo:** Conjunto de filas alternadas de un *cuadro*. Los *cuadros* constan de dos *campos*, el *campo superior* y el *campo inferior*.
- 3.45 macrobloque campo:** Macrobloque que contiene muestras de un solo *campo*. Todos los *macrobloques* de un campo codificado son *macrobloques campo*. Cuando se utiliza la *decodificación de cuadro/campo adaptativa por macrobloque*, es posible que algunos *macrobloques* del *cuadro codificado* sean *macrobloques campo*.
- 3.46 par de macrobloques campo:** *Par de macrobloques* decodificados como dos *macrobloques campo*.
- 3.47 barrido por campo:** Orden secuencial concreto de *coeficientes de transformada*; se diferencia del *barrido en zig-zag* en que las columnas se barren más rápidamente que las filas. Se utiliza para *coeficientes de transformada* en *macrobloques campo*.
- 3.48 bandera:** Variable con sólo dos valores posibles: 0 y 1.
- 3.49 cuadro:** Un *cuadro* contiene una matriz de muestras *luma* y sus dos matrices de muestras *croma* correspondientes. Cada *cuadro* consta de dos *campos*, un *campo superior* y un *campo inferior*.
- 3.50 macrobloque cuadro:** *Macrobloque* que representa muestras de los dos *campos* de un *cuadro codificado*. Si no se utiliza la *decodificación de cuadro/campo adaptativa por macrobloque*, todos los *macrobloques* del *cuadro codificado* son *macrobloques cuadro*. Si se utiliza la *decodificación de cuadro/campo adaptativa de macrobloque*, es posible que algunos *macrobloques* del *cuadro codificado* sean *macrobloques cuadro*.
- 3.51 par de macrobloques cuadro:** *Par de macrobloques* decodificados como dos *macrobloques cuadro*.
- 3.52 índice de frecuencia:** Índice monodimensional o bidimensional relacionado con un *coeficientes de transformada* antes de aplicar la *transformada inversa* del *proceso de decodificación*.

- 3.53 decodificador ficticio de referencia (HRD, hypothetical reference decoder):** Modelo de *decodificador* ficticio que especifica restricciones sobre la variabilidad de los *trenes de unidades NAL* conformes o de los *trenes de bytes* conformes que puede producir el proceso de codificación.
- 3.54 planificador ficticio de tren (HSS, hypothetical stream scheduler):** Mecanismo ficticio de suministro de la temporización y del tren de datos de la entrada de un *tren de bits* al *decodificador ficticio de referencia*. El HSS se utiliza para comprobar la conformidad de un *tren de bits* o de un *decodificador*.
- 3.55 sector I:** *Sector* que no es *sector SI* y que se decodifica con *predicción*, partiendo sólo de las muestras decodificadas dentro del mismo *sector*.
- 3.55.1 informativo:** Término utilizado para referirse a contenido provisto en esta Recomendación | Norma Internacional pero que no es parte integrante de esta Recomendación | Norma Internacional. El contenido informativo no implica exigencias obligatorias para la conformidad con esta Recomendación | Norma Internacional.
- 3.56 unidad de acceso de regeneración instantánea de decodificación (IDR, instantaneous decoding refresh):** *Unidad de acceso* cuya *imagen codificada primaria* es una *imagen IDR*.
- 3.57 imagen de regeneración de decodificación instantáneo:** *Imagen codificada* que sólo contiene *sectores* del tipo *I* o *SI*, lo que hace que el *proceso de decodificación* marque todas las *imágenes de referencia* como "no utilizada para referencia" inmediatamente después de la decodificación de la *imagen IDR*. Después de decodificar la *imagen IDR* se pueden decodificar todas las siguientes *imágenes codificadas* en el *orden de decodificación*, sin emplear la *predicción inter* de las *imágenes* decodificadas antes que la *imagen IDR*. La primera *imagen* de cada *secuencia de vídeo codificada* es una *imagen IDR*.
- 3.58 codificación inter:** Codificación de *bloque*, *macrobloque*, *sector*, o *imagen* que utiliza *predicción inter*.
- 3.59 predicción inter:** *Predicción* que se obtiene de las muestras decodificados de *imágenes de referencia* distintas de la *imagen decodificada* en cuestión.
- 3.60 codificación intra:** Codificación de *bloque*, *macrobloque*, *sector*, o *imagen* que utiliza *predicción intra*.
- 3.61 predicción intra:** *Predicción* que se obtiene de las muestras decodificadas del propio *sector decodificado*.
- 3.62 sector intra:** Véase sector I.
- 3.63 transformada inversa:** Parte del *proceso de decodificación* en la que el conjunto de *coeficientes de transformada* se convierte a valores en el dominio del espacio, o en la que el conjunto de *coeficientes de transformada* se convierte a *coeficientes de transformada c.c.*
- 3.64 capa:** Una estructura de un conjunto de estructuras sintácticas que mantienen una relación jerárquica no ramificada. Las capas superiores contienen a las inferiores. Las capas de codificación son las capas *secuencia vídeo codificado*, *imagen*, *sector*, y *macrobloque*.
- 3.65 nivel:** Conjunto definido de restricciones sobre los valores que pueden tener los *elementos sintácticos* y las variables en esta Recomendación | Norma Internacional. Se define el mismo conjunto de niveles para todos los *perfiles*, de modo que la mayoría de los aspectos de la definición de cada nivel son comunes a los diferentes *perfiles*. Es posible que cada implementación, dentro de las restricciones especificadas, tenga un nivel diferente para cada *perfil* que soporta. En un contexto diferente, un *nivel* es el valor de un *coeficiente de transformada* antes del *cambio de escala*.

- 3.66 vector de movimiento de lista 0 (lista 1):** *Vector de movimiento* relacionado con un *índice de referencia* que apunta a *lista 0 (lista 1) de imagen de referencia*.
- 3.67 predicción lista 0 (lista 1):** *Predicción inter* del contenido de un *sector* mediante un *índice de referencia* que apunta a un *lista 0 (lista 1) de imagen de referencia*.
- 3.68 luma:** Adjetivo que califica a una muestra o matriz de muestras que representa la señal monocromática relativa a los colores primarios. El símbolo o subíndice utilizado para luma es Y o L.  
 NOTA – Se utiliza el término "luma" en lugar de "luminancia" porque este último se asocia con la utilización de características de transferencia de luz lineal. A veces se emplea el símbolo L a cambio del símbolo Y para evitar confusión con el símbolo y que se utiliza para denotar la posición vertical.
- 3.69 macrobloque:** *Bloque* de 16x16 de muestras *luma* y sus dos *bloques* correspondientes de muestras *croma*. La división de un *sector* o de un *par de macrobloques* en macrobloques se denomina *subdivisión en particiones*.
- 3.70 decodificación de cuadro/campo adaptativa por macrobloque:** *Proceso de decodificación* de *cuadros codificados* en el cual algunos *macrobloques* pueden estar decodificados como *macrobloques cuadro* y otros como *macrobloques campo*.
- 3.71 dirección del macrobloque:** Cuando no se utiliza la *decodificación de cuadro/campo adaptativa por macrobloque*, la dirección del macrobloque es el índice del macrobloque de un *barrido por filas del macrobloque* de la *imagen* contando desde cero para el *macrobloque superior izquierdo* de la *imagen*. Cuando se utiliza la *decodificación de cuadro/campo adaptativa de macrobloque*, la dirección del macrobloque del *macrobloque superior* de un *par de macrobloques* es el doble del índice del *par de macrobloques* de un *barrido por filas del macrobloque* de la *imagen*, y la dirección del macrobloque del *macrobloque inferior* de un *par de macrobloques* es la dirección del macrobloque del correspondiente *macrobloque superior* más 1. La dirección del macrobloque del *macrobloque superior* de cada *par de macrobloques* es un número par, y la dirección del macrobloque del *macrobloque inferior* de cada *par de macrobloques* es un número impar.
- 3.72 posición del macrobloque:** Las coordenadas bidimensionales de un *macrobloque* en una *imagen* representadas por ( x, y ). El *macrobloque superior izquierdo* de la *imagen* ( x, y ) vale ( 0, 0 ). x se incrementa en 1 para cada columna del *macrobloque* de izquierda a derecha. Cuando no se utiliza la *decodificación de cuadro/campo adaptativa de macrobloque*, "y" se incrementa en 1 para cada fila del *macrobloque* de arriba abajo. Cuando se utiliza la *decodificación de cuadro/campo adaptativa de macrobloque*, "y" se incrementa en 2 para cada fila del *par de macrobloques* de arriba abajo, y se incrementa una unidad más cuando el macrobloque es un *macrobloque inferior*.
- 3.73 par de macrobloques:** Par de *macrobloques* de un *cuadro*, contiguos verticalmente y apareados para la *decodificación de cuadro/campo adaptativa de macrobloque*. La división de un *sector* en pares de macrobloques se denomina *subdivisión en particiones*.
- 3.74 partición macrobloque:** *Bloque* de muestras *luma* y sus dos *bloques* de muestras *croma* correspondientes que resultan de *subdividir en particiones* un *macrobloque* para realizar *predicción Inter*.
- 3.75 mapeado (o correspondencia) de macrobloques en grupos de sectores:** Mapeado de *macrobloques* de una *imagen* en *grupos de sectores*. Consta de una lista de números, una por cada *macrobloque* codificado, que especifica el *grupo de sectores* al cual pertenece cada *macrobloque* codificado.
- 3.76 mapeado de unidades de mapeado (o correspondencia) en grupos de sectores:** Una forma de mapear *unidades de mapeado de grupos de sectores* de una *imagen* en *grupos de sectores*. Consta de una lista de números, una para cada *unidad de mapeado de grupos de*

*sectores*, que especifica el *grupo de sectores* al que pertenece cada *unidad de mapeado de grupos de sectores*.

- 3.76.1 puede:** Término utilizado para referirse a un comportamiento que aunque está permitido no es necesariamente exigido. En algunos casos en los que se desea hacer énfasis en la naturaleza opcional del comportamiento descrito se emplea la frase "puede o no".
- 3.77 operación control de gestión de memoria:** Siete operaciones que controlan el *marcado de la imagen de referencia*.
- 3.78 vector de movimiento:** Vector bidimensional utilizado en la *predicción inter*, que indica una traslación desde las coordenadas de la *imagen decodificada* a las coordenadas de la *imagen de referencia*.
- 3.78.1 debe:** Se utiliza para expresar una observación acerca de un requisito o una implicación de un requisito especificado en alguna otra parte de esta Recomendación | Norma Internacional. Este término se utiliza exclusivamente en un contexto *informativo*.
- 3.79 unidad NAL:** Estructura sintáctica que indica el tipo de datos siguientes y los bytes de los propios datos en forma de una *RBSP* y, cuando sea necesario, con *bytes de prevención de emulación* intercalados.
- 3.80 tren de unidades NAL:** Secuencia de *unidades NAL*.
- 3.81 campo desapareado:** Nombre genérico que designa un campo de *referencia desapareado* o un campo *que no es de referencia desapareado*.
- 3.82 campo desapareado que no es de referencia:** *Campo que no es de referencia* codificado, que no forma parte de un *par de campos complementarios distintos de los de referencia*.
- 3.83 campo de referencia desapareado:** *Campo de referencia* decodificado, que no forma parte de un *par de campos complementarios de referencia*.
- 3.84 campo que no es de referencia:** *Campo codificado*, cuyo *nal\_ref\_idc* es igual a 0.
- 3.85 cuadro que no es de referencia:** *Cuadro* codificado, cuyo *nal\_ref\_idc* es igual a 0.
- 3.86 imagen que no es de referencia:** *Imagen* codificada, cuyo *nal\_ref\_idc* es igual a 0. Las *imágenes distintas de las de las de referencia* no se utilizan para la *predicción inter* de otras *imágenes*.
- 3.86.1 nota:** Término empleado para prefijar observaciones *informativas*. Este término se utiliza exclusivamente en un contexto *informativo*.
- 3.87 paridad contraria:** La *paridad contraria* de *superior* es *inferior*, y viceversa.
- 3.88 orden de salida:** Orden en que salen las *imágenes decodificadas* de la *memoria intermedia de imágenes decodificadas*.
- 3.89 sector P:** *Sector* que se puede decodificar por *predicción intra* de muestras decodificadas dentro del mismo *sector* o por *predicción inter* de *imágenes de referencia* previamente decodificadas, de modo que para *predecir* los valores de las muestras de cada *bloque* se utiliza a lo sumo un *vector de movimiento* y un *índice de referencia*.
- 3.90 parámetro:** *Elemento sintáctico* de un *conjunto de parámetros secuencia* o de un *conjunto de parámetros imagen*. El término "parámetro" también se utiliza en la definición del término *parámetro cuantificación*.
- 3.91 paridad:** La paridad de un *campo* puede ser *superior* o *inferior*.
- 3.92 subdivisión en particiones:** División de un conjunto en subconjuntos de modo que cada elemento del conjunto esté en sólo uno de los subconjuntos.
- 3.93 imagen:** Nombre genérico que designa un *campo* o un *cuadro*.

- 3.93.1 conjunto de parámetros de imagen:** *Estructura sintáctica* que contiene *elementos sintácticos* aplicables a cero o más *imágenes codificadas* completas según lo determina el *elemento sintáctico* `pic_parameter_set_id` que se encuentra en cada *cabecera de sector*.
- 3.94 número de orden de la imagen:** Variable cuyo valor no disminuye al aumentar la posición de la *imagen* en el orden de salida con respecto a la *imagen IDR* anterior en el *orden de decodificación* o con respecto a la *imagen* anterior que contiene la *operación control de gestión de memoria* que marca todas las *imágenes de referencia* con "no utilizada para referencia".
- 3.95 predicción:** Concretización de un *proceso de predicción*.
- 3.96 proceso de predicción:** Cálculo mediante un *predictor* del valor de la muestra o del elemento de datos que se está decodificando.
- 3.97 sector predictivo:** Véase sector P.
- 3.98 predictor:** Combinación de valores especificados o de valores de muestras o elementos de datos previamente decodificados que se utilizan en el *proceso de decodificación* de los valores de muestra o elementos de datos subsiguientes.
- 3.99 imagen codificada primaria:** Representación codificada de una *imagen* que se utiliza en el *proceso de decodificación* de un tren de bits conforme con esta Recomendación | Norma Internacional. La imagen codificada primaria contiene todos los *macrobloques* de la *imagen*. Las únicas *imágenes* que tienen un efecto normativo en el *proceso de decodificación* son las imágenes codificada primarias. Véase también *imagen codificada redundante*.
- 3.100 perfil:** Un subconjunto determinado de la sintaxis de esta Recomendación | Norma Internacional.
- 3.101 parámetro cuantificación:** Variable utilizada en el *proceso de decodificación* para el *cambio de escala de niveles de coeficiente de transformada*.
- 3.102 acceso aleatorio:** Inicio del *proceso de decodificación* de un *tren de bits* en un punto distinto del principio del mismo.
- 3.103 barrido por filas:** Mapeado de un patrón bidimensional rectangular en un patrón unidimensional de modo que las primeras muestras del patrón unidimensional correspondan a la primera fila de arriba del patrón bidimensional barrido de izquierda a derecha, seguido de la segunda, tercera, etc., filas del patrón (de arriba abajo), y barridas de izquierda a derecha.
- 3.104 cabida útil de secuencia de bytes en bruto (RBSP, raw byte sequence payload):** Estructura sintáctica que contiene un número entero de *bytes* encapsulada en una *unidad NAL*. La RBSP puede estar vacía o tener una *cadena de bits de datos* con *elementos sintácticos* seguido de un *bit de paro RBSP* y de uno o varios bits de valor 0.
- 3.105 bit de paro de la cabida útil de secuencia de bytes en bruto:** Bit de valor 1 en una *cabida útil de secuencia de bytes en bruto (RBSP)* después de una *cadena de bits de datos*. Es posible determinar el fin de la *cadena de bits de datos* en una *RBSP* buscando el *bit de paro RBSP* desde el final de la *RBSP*, que es el último bit distinto de cero en la *RBSP*.
- 3.106 punto de recuperación:** Punto del *tren de bits* a partir del cual se logra una representación exacta o aproximada de las *imágenes decodificadas* representadas por el *tren de bits* después de un *acceso aleatorio* o un *enlace irregular*.
- 3.107 imagen codificada redundante:** Representación codificada de una *imagen* o de parte de una *imagen*. El contenido de una imagen codificada redundante no se utiliza en el *proceso de decodificación* de un *tren de bits* conforme con esta Recomendación | Norma



Internacional. No es obligatorio que la *imagen codificada redundante* contenga todos los *macrobloques* de la *imagen codificada primaria*. Las imágenes codificadas redundantes no tiene efecto normativo en el *proceso de decodificación*. Véase también *imagen codificada primaria*.

- 3.108 campo de referencia:** Es posible utilizar los campos *de referencia* para la *predicción inter* cuando se decodifican *sectores P, SP, y B* de un *campo codificado* o de *macrobloques campo* de un *cuadro codificado*. Véase también *imagen de referencia*.
- 3.109 cuadro de referencia:** Es posible utilizar los *cuadros de referencia* para la *predicción inter* cuando se decodifican *sectores P, SP, y B* de un *campo codificado*. Véase también *imagen de referencia*.
- 3.110 índice de referencia:** Índice de una *lista de imágenes de referencia*.
- 3.111 imagen de referencia:** *Imagen* con *nal\_ref\_idc* distinto de 0. Las *imágenes de referencia* contienen muestras que se pueden utilizar para la *predicción inter* en el *proceso de decodificación* de las subsiguientes *imágenes en orden de decodificación*.
- 3.112 lista de imágenes de referencia:** Lista de números de *imagen* de corto y largo alcance que se asigna a las *imágenes de referencia*.
- 3.113 lista 0 de imágenes de referencia:** *Lista de imágenes de referencia* utilizada para la *predicción inter* de un *sector P, B, o SP*. Toda *predicción inter* utilizada para los sectores *P y SP* utiliza la lista 0 de imágenes de referencia. La lista 0 de imágenes de referencia es una de las dos *listas de imágenes de referencia* que se utilizan en la *predicción inter* de un *sector B*, siendo la otra la *lista 1 de imágenes de referencia*.
- 3.114 lista 1 de imágenes de referencia:** *Lista de imágenes de referencia* utilizadas para la *predicción inter* de un *sector B*. La lista 1 de imágenes de referencia es una de las dos *listas de imágenes de referencia* que se utilizan en la *predicción inter* de un *sector B*, siendo la otra la *lista 0 de imágenes de referencia*.
- 3.115 marcado de imágenes de referencia:** Especifica, en el tren de bits, cómo se marcan las *imágenes decodificadas* para la *predicción inter*.
- 3.116 reservado:** Cuando se utiliza este término en las cláusulas que especifican valores de un determinado *elemento sintáctico*, significa que el valor está reservado para un uso futuro en UIT-T | ISO/CEI. Si bien estos valores no se utilizan en los *trenes de bits* conformes con esta Recomendación | Norma Internacional, es posible que se utilicen en futuras extensiones de esta Recomendación | Norma Internacional del UIT-T | ISO/CEI.
- 3.117 residuo:** Diferencia entre la *predicción* de una muestra o un elemento de datos y su valor decodificado.
- 3.118 serie:** Varios elementos de datos consecutivos representados en el proceso de decodificación. En un contexto, se aplica al número de *niveles de coeficientes de transformada* de valor cero que preceden al *nivel de coeficientes de transformada* distintos de cero en la lista de *niveles de coeficientes de transformada* generados por un *barrido en zig-zag* o un *barrido por campo*. En otros contextos, este término se utiliza para referirse a una serie de *macrobloques*.
- 3.119 relación de aspecto de muestras:** Especifica la relación entre la distancia horizontal deseada entre columnas y la distancia vertical deseada entre filas de la matriz de muestras *luma* en un *cuadro*, y sirve de ayuda la visualización, que no se especifica en esta Recomendación | Norma Internacional. La relación de aspecto de muestra se expresa como *h:v*, siendo *h* la anchura y *v* la altura (en cualquier unidad de distancia espacial).
- 3.120 cambio de escala:** Proceso de multiplicar los *niveles de coeficientes de transformada* por un factor, que da por resultado *coeficientes de transformada*.

- 3.120.1 conjunto de parámetros de secuencia:** *Estructura sintáctica* que contiene *elementos sintácticos* que se aplican a cero o más *secuencias de vídeo codificado* completas según lo determina el contenido de un *elemento sintáctico* `seq_parameter_set_id` que se encuentra en el *conjunto de parámetros de imagen* al que hace referencia el *elemento sintáctico* `pic_parameter_set_id` encontrado en cada *cabecera de sector*.
- 3.120.2 deberá (tiempo futuro de mandato):** Se utiliza para expresar requisitos obligatorios para la observancia de esta Recomendación | Norma Internacional. Cuando se utiliza para expresar exigencias obligatorias de los valores de *elementos sintácticos* o de los resultados obtenidos mediante el empleo del *proceso de decodificación* específico, es responsabilidad del *codificador* asegurarse de que se cumplan las exigencias. Cuando se utiliza con relación a las operaciones desarrolladas por el *proceso de decodificación*, cualquier *proceso de decodificación* que produzca resultados idénticos al *proceso de decodificación* aquí descrito se ajusta a los requisitos del *proceso de decodificación* de esta Recomendación | Norma Internacional.
- 3.120.3 debería:** Se utiliza para referirse al comportamiento de una implementación cuyo desarrollo se alienta bajo circunstancias ordinarias previstas, pero sin ser una exigencia obligatoria para la observancia de esta Recomendación | Norma Internacional.
- 3.121 sector SI:** *Sector* que está codificado mediante *predicción* sólo a partir de muestras decodificadas del mismo *sector* y con cuantificación de las muestras de *predicción*. Los sectores SI se pueden codificar de manera que sus muestras decodificadas pueda estructurarse como si fueran un *sector SP*.
- 3.122 macrobloque obviado:** *Macrobloque* sin datos codificados, salvo una indicación de que se ha de decodificar como "obviado". Esta indicación puede ser común a varios *macrobloques*.
- 3.123 sector:** Número entero de *macrobloques* o *pares de macrobloques* ordenados consecutivamente en *barrido por filas* dentro de un determinado *grupo de sectores*. Para la *imagen codificada primaria*, la división en sectores de cada *grupo de sectores* es una *subdivisión en particiones*. Aunque es posible que un sector contenga *macrobloques* o *pares de macrobloques* que sean consecutivos en el barrido por filas dentro de un grupo de sectores, estos *macrobloques* o *pares de macrobloques* no son necesariamente consecutivos en el barrido por filas dentro de la *imagen*. Las direcciones de los *macrobloques* se obtienen de la dirección del primer *macrobloque* de un sector (como aparece en la *cabecera de sector*) y del *mapeado del macrobloque en el grupo de sectores*.
- 3.124 subdivisión en particiones de datos de sector:** Método de *subdividir en particiones* ciertos *elementos sintácticos* en *estructuras sintácticas* basadas en una *categoría* asociada con cada *elemento sintáctico*.
- 3.125 grupo de sectores:** Subconjunto de *macrobloques* o *pares de macrobloques* de una *imagen*. La división de la *imagen* en grupos de sectores es una *subdivisión en particiones* de la *imagen*. La subdivisión en particiones se especifica en el *mapeado de macrobloques en grupo de sectores*.
- 3.126 unidades de mapeado (o correspondencia) de grupos de sectores:** *Unidades de mapeado* de la *unidad de mapeado en grupo de sectores*.
- 3.127 cabecera de sector:** Parte del *sector codificado* que contiene los elementos de datos que pertenecen al primero o a todos los *macrobloques* representados en el sector.
- 3.128 fuente:** Término que se utiliza para describir el material vídeo o alguno de sus atributos antes de la codificación.
- 3.129 sector SP:** *Sector* que se codifica mediante *predicción inter* de *imágenes de referencia* decodificadas previamente, con a lo sumo un *vector de movimiento* y un *índice de*

*referencia* para *predecir* los valores de las muestras de cada *bloque*. El sector SP se puede codificar de forma que sus muestras decodificadas se puedan estructurar de la misma manera que otro sector SP o un *sector SI*.

- 3.130 prefijo código de inicio:** Secuencia única de tres bytes, cuyo valor es 0x000001 incluida en el *tren de bytes* y que es el prefijo de cada *unidad NAL*. El *decodificador* puede utilizar la posición del *prefijo código de inicio* para determinar el comienzo de una nueva *unidad NAL* y el fin de la *unidad NAL* anterior. Para evitar la emulación de prefijos código de inicio en *unidades NAL* se incluyen *bytes de prevención de emulación*.
- 3.131 cadena de bits de datos (SODB, string of data bits):** Secuencia de varios bits que representa *elementos sintácticos* dentro de una *cabida útil de secuencia de bytes en bruto* y que aparecen antes que el *bit de paro de la cabida útil de secuencia de bytes en bruto*. En una *SODB*, se considera que el bit más a la izquierda es el primero y el más significativo, y el más a la derecha, el último y menos significativo.
- 3.132 submacrobloque:** La cuarta parte de las muestras de un *macrobloque*, es decir, un *bloque luma* de 8x8 y dos *bloques croma* correspondientes, de modo que una de sus esquinas coincide con una esquina del *macrobloque*.
- 3.133 partición submacrobloque:** *Bloque* de muestras *luma* y sus dos *bloques* de muestras *croma* correspondientes, que resulta de *subdividir en particiones* un *submacrobloque* para la *predicción inter*.
- 3.134 intercambio de sector I:** Véase sector SI.
- 3.135 intercambio de sector P:** Véase sector SP.
- 3.136 elemento sintáctico:** Elemento de datos representado en el *tren de bits*.
- 3.137 estructura sintáctica:** *Elementos sintácticos* que aparecen juntos en el *tren de bits* y en un determinado orden.
- 3.138 campo superior:** Una de las dos *campos* que forman un *cuadro*. Cada fila de un *campo superior* está situada justo debajo de la correspondiente fila del *campo inferior*.
- 3.139 macrobloque superior (de un par de macrobloques):** *Macrobloque* dentro de un *par de macrobloques* que contiene las muestras de la fila superior de muestras del *par de macrobloques*. En un *par de macrobloques campo*, el macrobloque superior representa las muestras de la región del *campo superior* del *cuadro* que está dentro de la región espacial del *par de macrobloques*. En un *par de macrobloques cuadro*, el macrobloque superior representa las muestras del *cuadro* que está en la mitad superior de la región espacio del *par de macrobloques*.
- 3.140 coeficiente de transformada:** Magnitud escalar, considerada en el dominio de la frecuencia, asociada a un determinado *índice de frecuencia* unidimensional o bidimensional en una *transformada inversa* del *proceso de decodificación*.
- 3.141 nivel de coeficiente de transformada:** Magnitud entera que representa el valor asociado con un determinado índice de frecuencia bidimensional en el *proceso de decodificación* antes del *cambio de escala* para calcular el valor del *coeficiente de transformada*.
- 3.142 identificador único universal (UUID, universal unique identifier):** Identificador que es único en el espacio de todos los identificadores únicos universales.
- 3.143 no especificado:** Cuando se utiliza este término en las cláusulas que especifican valores de un determinado *elemento sintáctico*, significa que los valores no tienen un significado en esta Recomendación | Norma Internacional y que no tendrán un significado específico en el futuro como parte integrante de esta Recomendación | Norma Internacional.

- 3.144** **codificación de longitud variable (VLC, *variable length coding*)**: Procedimiento reversible de codificación entrópica que asigna las cadenas de bits más cortas a los *símbolos* que se prevé sean más frecuentes, y las cadenas de bits más largas a los *símbolos* que se prevé sean menos frecuentes.
- 3.145** **barrido en zig-zag**: Determinado orden secuencial de *niveles de coeficientes de transformada* que va (aproximadamente) de la frecuencia espacial menor a la mayor. El barrido en zig-zag se utiliza para *niveles de coeficiente de transformada* en *macrobloques cuadro*.

#### **4 Abreviaturas, siglas o acrónimos**

- 4.1** **CABAC**: Codificación aritmética binaria adaptativa basada en el contexto (*context-based adaptive binary arithmetic coding*)
- 4.2** **CAVLC**: Codificación de longitud variable adaptativa basada en el contexto (*context-based adaptive variable length coding*)
- 4.3** **CBR**: Velocidad binaria constante (*constant bit rate*)
- 4.4** **CPB**: Memoria intermedia de imagen codificada (*coded picture buffer*)
- 4.5** **DPB**: Memoria intermedia de imagen decodificada (*decoded picture buffer*)
- 4.6** **DUT**: Decodificador en prueba (*decoder under test*)
- 4.7** **FIFO**: Primero en entrar, primero en salir (*first-in, first-out*)
- 4.8** **HRD**: Decodificador ficticio de referencia (*hypothetical reference decoder*)
- 4.9** **HSS**: Planificador ficticio de tren (*hypothetical stream scheduler*)
- 4.10** **IDR**: Regeneración de decodificación instantánea (*instantaneous decoding refresh*)
- 4.11** **LSB**: Bit menos significativo (*least significant bit*)
- 4.12** **MB**: Macrobloque (*macroblock*)
- 4.13** **MBAFF**: Codificación de cuadro-campo adaptativa por macrobloque (*macroblock-adaptive frame-field coding*)
- 4.14** **MSB**: Bit más significativo (*most significant bit*)
- 4.15** **NAL**: Capa de abstracción de red (*network abstraction layer*)
- 4.16** **RBSP**: Cabida útil de secuencia de bytes en bruto (*raw byte sequence payload*)
- 4.17** **SEI**: Información de perfeccionamiento complementaria (*supplemental enhancement information*)
- 4.18** **SODB**: Cadena de bits de datos (*string of data bits*)
- 4.19** **UUID**: Identificador único universal (*universal unique identifier*)
- 4.20** **VBR**: Velocidad binaria variable (*universal unique identifier*)
- 4.21** **VCL**: Capa de codificación de vídeo (*video coding layer*)
- 4.22** **VLC**: Codificación de longitud variable (*variable length coding*)
- 4.23** **VUI**: Información de utilización de vídeo (*video usability information*)

## 5 Convenios

NOTA – Los operadores matemáticos utilizados en esta Especificación son similares a los del lenguaje de programación C. Además se definen los operadores de división entera y desplazamiento aritmético. Por convenio, al enumerar y contar se empieza generalmente desde 0.

### 5.1 Operadores aritméticos

Se definen los siguientes operadores aritméticos:

- + Suma
- Resta (cuando se especifican dos argumentos) o negación (como operador de prefijo unario)
- \* Multiplicación
- $x^y$  Exponencial. Significa  $x$  a la potencia  $y$ . En otros contextos esta notación se utiliza para indicar un superíndice, y no se ha de interpretar como una exponencial.
- / División entera en la que se descartan los decimales. Ejemplo,  $7/4$  y  $-7/-4$  valen 1 y  $-7/4$  y  $7/-4$  valen  $-1$ .
- ÷ División, se utiliza en ecuaciones matemáticas y no se descartan los decimales ni se redondea.
- $\frac{x}{y}$  División, se utiliza en ecuaciones matemáticas y no se descartan los decimales ni se redondea.
- $\sum_{i=x}^y f(i)$  Sumatorio de  $f(i)$ , donde  $i$  toma todos los valores enteros comprendidos entre  $x$  e  $y$  inclusive.
- $x \% y$  Módulo. Se obtiene el resto de dividir  $x$  por  $y$ ; está definido sólo para valores enteros de  $x$  e  $y$ , siendo  $x \geq 0$  e  $y > 0$ .

Si no se utilizan paréntesis para indicar explícitamente el orden de precedencia, se aplican las siguientes reglas:

- la multiplicación y la división se realizan antes que la suma y la resta;
- la multiplicación y la división en una expresión se calculan por orden de izquierda a derecha;
- la suma y la resta en una expresión se calculan por orden de izquierda a derecha.

### 5.2 Operadores lógicos

Se definen los siguientes operadores lógicos

- $x \ \&\& \ y$  función lógica booleana "and" entre  $x$  e  $y$
- $x \ || \ y$  función lógica booleana "or" entre  $x$  e  $y$
- ! función lógica booleana "not"
- $x \ ? \ y \ : \ z$  si  $x$  es TRUE (verdadero) o distinto de 0, toma el valor de  $y$ ; en caso contrario, toma el valor de  $z$ .

### 5.3 Operadores relacionales

Se definen los siguientes operadores relacionales

- > mayor que
- >= mayor o igual que

- < menor que
- <= menor o igual que
- = = igual a
- != distinto de.

## 5.4 Operadores de bit

Se definen los siguientes operadores de bit:

- & Función lógica "and" bit a bit. Cuando se aplica a números enteros, se toma la representación en complemento a dos de los números enteros. Cuando se aplica entre dos números binarios con distinto número de bits, al más corto se le añaden ceros a la izquierda, se rellenan con 0 los bits más significativos.
- | Función lógica "or" bit a bit. Cuando se aplica a números enteros, se toma la representación en complemento a dos de los números enteros. Cuando se aplica entre dos números binarios con distinto número de bits, al más corto se le añaden ceros a la izquierda, se rellenan con 0 los bits más significativos.
- x >> y Desplaza y posiciones hacia la derecha los bits de x, estando x expresado en complemento a dos. Esta función sólo está definida para valores enteros positivos de y. El valor de los MSB de x es el mismo antes y después del desplazamiento.
- x << y Desplaza y posiciones hacia la izquierda los bits de x, estando x expresado en complemento a dos. Esta función sólo está definida para valores enteros positivos de y. Al aplicar el desplazamiento, los bits añadidos en los LSB tiene un valor igual a 0.

## 5.5 Operadores de asignación

Se definen los siguientes operadores de asignación:

- = Operador de asignación.
- ++ Aumento: x++ significa  $x = x + 1$ ; cuando se utiliza en un índice de una matriz, el valor de la variable se toma antes de aplicar el operador aumento.
- Disminución: x-- significa  $x = x - 1$ ; cuando se utiliza en un índice de una matriz, el valor de la variable se toma antes de aplicar el operador disminución.
- += Aumento por un valor determinado: x += 3 significa  $x = x + 3$ , y x += (-3) significa  $x = x + (-3)$ .
- = Disminución por un valor determinado: x -= 3 significa  $x = x - 3$ , y x -= (-3) significa  $x = x - (-3)$ .

## 5.6 Notación para indicar una gama de valores

Se utiliza la siguiente notación para indicar una gama de valores

- x = y .. z x puede tener cualquier valor comprendido entre "y" y "z" inclusive, siendo x, y, z números enteros.

## 5.7 Funciones matemáticas

Se definen las siguientes funciones matemáticas:

$$\text{Abs}(x) = \begin{cases} x & ; x \geq 0 \\ -x & ; x < 0 \end{cases} \quad (5-1)$$

$$\text{Ceil}(x) \quad \text{calcula el entero más pequeño que es mayor o igual a } x. \quad (5-2)$$

$$\text{Clip1}(x) = \text{Clip3}(0, 255, x) \quad (5-3)$$

$$\text{Clip3}(x, y, z) = \begin{cases} x & ; z < x \\ y & ; z > y \\ z & ; \text{otros} \end{cases} \quad (5-4)$$

$$\text{Floor}(x) \quad \text{calcula el entero más grande que es menor o igual a } x. \quad (5-5)$$

$$\text{InverseRasterScan}(a, b, c, d, e) = \begin{cases} (a \% (d/b)) * b; & e == 0 \\ (a / (d/b)) * c; & e == 1 \end{cases} \quad (5-6)$$

$$\text{Log2}(x) \quad \text{calcula el logaritmo en base 2 de } x \quad (5-7)$$

$$\text{Log10}(x) \quad \text{calcula el logaritmo en base 10 de } x \quad (5-8)$$

$$\text{Luma4x4BlkScan}(x, y) = (x / 2) * 4 + (y / 2) * 8 + \text{RasterScan}(x \% 2, y \% 2, 2) \quad (5-9)$$

$$\text{Median}(x, y, z) = x + y + z - \text{Mín}(x, \text{Mín}(y, z)) - \text{Máx}(x, \text{Máx}(y, z)) \quad (5-10)$$

$$\text{Mín}(x, y) = \begin{cases} x & ; x \leq y \\ y & ; x > y \end{cases} \quad (5-11)$$

$$\text{Máx}(x, y) = \begin{cases} x & ; x \geq y \\ y & ; x < y \end{cases} \quad (5-12)$$

$$\text{RasterScan}(x, y, n_x) = x + y * n_x \quad (5-13)$$

$$\text{Round}(x) = \text{Sign}(x) * \text{Floor}(\text{Abs}(x) + 0,5) \quad (5-14)$$

$$\text{Sign}(x) = \begin{cases} 1 & ; x \geq 0 \\ -1 & ; x < 0 \end{cases} \quad (5-15)$$

$$\text{Sqrt}(x) = \sqrt{x} \quad (5-16)$$

## 5.8 Variables, elementos sintácticos y cuadros

Los elementos sintácticos en el tren de bits se representan en **negritas**. Cada elemento sintáctico se describe por su nombre (en minúsculas y con el carácter subrayado), su categoría o sus dos categorías sintácticas, y uno o dos descriptores de su método de representación codificada. El proceso de decodificación se lleva a cabo de acuerdo con el valor del elemento sintáctico y de los valores de los elementos sintácticos decodificados previamente. Cuando el valor de un elemento sintáctico se utiliza en los cuadros de sintaxis o en el texto, aparece en letra normal (es decir, no está en negritas).

Quizá en algunos casos los cuadros sintácticos utilicen los valores de otras variables derivadas de los valores de los segmentos sintácticos. Estas variables aparecen en los cuadros de sintaxis o en el texto en una combinación de mayúsculas y minúsculas y sin el carácter subrayado. Las variables que empiecen con mayúscula se obtienen para la decodificación de la estructura sintáctica considerada y todas las estructuras sintácticas dependientes. Las variables que empiezan con mayúscula se pueden utilizar en el proceso de decodificación de estructuras sintácticas posteriores que mencionen la estructura sintáctica original de la variable. Las variables que empiezan con minúscula sólo se utilizan en la subcláusula en que tiene su origen.

En algunos casos, se utilizan indistintamente nombres "nematécnicos" o sus valores numéricos para referirse a los valores de los elementos sintácticos o valores variables. Algunas veces se utilizan nombres "nematécnicos" que no tienen asociados valores numéricos. La relación entre valores y nombres se especifican en el texto. Los nombres están formados por un grupo de letras o varios grupos separados por un carácter subrayado. Cada grupo empieza con mayúscula aunque puede contener varias letras mayúsculas.

NOTA – La sintaxis se describe de manera muy similar a la estructura sintáctica del lenguaje C.

Las funciones se describen mediante un nombre, cuya forma es la de los nombres de elementos sintácticos, y entre paréntesis encierra los nombres de las variables (para su definición) o valores (para su utilización), separados por comas (en caso de que haya más de una variable).

Los corchetes se utilizan para la indexación de listas o matrices. Éstas pueden ser elementos sintácticos o variables. A veces las matrices bidimensionales se especifican también mediante una notación matricial cuyos índices se indican mediante subíndices.

NOTA – El orden de los índices de las matrices bidimensionales que utilizan corchetes y subíndices es intercambiable. Una muestra cuya posición horizontal es x, y cuya posición vertical es y, que en una matriz de muestras bidimensional se indica mediante  $s[x, y]$  se podría representar en notación matricial como  $s_{yx}$ .

La notación binaria se indica poniendo la cadena de bits entre comillas sencillas. Por ejemplo, '01000001' representa una cadena de ocho bits en la que sólo vale 1 el segundo bit y el último.

Cuando el número de bits es un múltiplo entero de cuatro, en lugar de la notación binaria se puede utilizar notación hexadecimal, que se indica mediante el prefijo "0x", antes del número hexadecimal. Por ejemplo, 0x41 representa una cadena de ocho bits en la que sólo vale 1 el segundo bit y el último.

Los valores numéricos que no están entre comillas sencillas y que no tienen el prefijo "0x" son valores decimales.

El valor 0 representa la condición FALSE (FALSO) en una expresión condicional. El valor TRUE (VERDADERO) se representa por cualquier otro valor distinto de cero.

## 5.9 Descripción mediante texto de operaciones lógicas

La expresión de operaciones lógicas que en pseudocódigo se describirían como:

```
if( condición 0 )
    expresión 0
else if ( condición 1 )
    expresión 1
...
else /* observación sobre las condiciones restantes */
    expresión n
```

en el texto se escribe del modo siguiente:

... del modo siguiente /... se aplica lo siguiente.

- Si condición 0, expresión 0
- De lo contrario, si condición 1, expresión 1
- ...
- De lo contrario (observación sobre las condiciones restantes), expresión n

Cada expresión "Si...de lo contrario, si...de lo contrario, ..." en el texto empieza con "... del modo siguiente" o "... se aplica lo siguiente" seguido inmediatamente de "Si ...". La última condición del "Si...de lo contrario, si... de lo contrario, ..." siempre es un "De lo contrario, ...". El intercalado de expresiones "Si...de lo contrario, si...de lo contrario, ..." se puede identificar por que empieza con "... del modo siguiente" o "... se aplica lo siguiente" y termina con el último "De lo contrario, ...".

La expresión de operaciones lógicas que en pseudocódigo se describirían mediante:

```
if( condición 0a && condición 0b )
    expresión 0
else if (condición 1a || condición 1b )
    expresión 1
...
```



else  
expresión n

en el texto se escribe del siguiente modo:

... del modo siguiente / ... se aplica lo siguiente.

- Si se cumplen todas las condiciones siguientes, expresión 0
  - condición 0a
  - condición 0b
- De lo contrario, si se cumple alguna de las siguientes condiciones, expresión 1
  - condición 1a
  - condición 1b
- ...
- De lo contrario, expresión n

La expresión de operaciones lógicas que en pseudocódigo se escriben como

```
if( condición 0 )  
    expresión 0  
if( condición 1 )  
    expresión 1
```

en el texto se escriben de la siguiente manera:

```
si condición 0, expresión 0  
si condición 1, expresión 1
```

## 5.10 Procesos

Los procesos se utilizan para describir la decodificación de elementos sintácticos. La especificación y la activación de un proceso son independientes. Todos los elementos sintácticos y las variables en mayúsculas que pertenecen a la estructura sintáctica en cuestión y las estructuras sintácticas dependientes están disponibles en la especificación y la activación del proceso. Dicha especificación puede tener además una variable en minúsculas que se especifica explícitamente como entrada o argumento. Cada especificación de proceso tiene especificada explícitamente una salida o resultado que es una variable que puede representarse en minúsculas o en mayúsculas.

La asignación de variables se especifica del modo siguiente:

- Cuando se activa un proceso, a las variables se les asignan explícitamente variables de entrada o salida en minúsculas de la especificación del proceso, en caso de que no tengan el mismo nombre.
- De lo contrario (cuando las variables en la activación y la especificación del proceso tienen el mismo nombre), la asignación es implícita.

En la especificación de un proceso, se puede hacer referencia a un macrobloque particular mediante el nombre de la variable cuyo valor es la dirección del macrobloque en cuestión.

## 6 Formato de los datos fuente, codificados, decodificados y de salida, proceso de barrido y relaciones de adyacencia

### 6.1 Formato del tren de bits

Esta subcláusula especifica la relación entre el tren de unidades NAL y el tren de bytes, a los cuales se hace referencia como tren de bits.

El tren de bits puede tener uno de los dos formatos siguientes: formato tren de unidades NAL o formato tren de bytes. Conceptualmente, el formato tren de unidades NAL es el tipo más "básico". Está formado por una secuencia de estructuras sintácticas denominadas unidades NAL. La secuencia está ordenada por orden de decodificación. Se imponen restricciones al orden de decodificación (y su contenido) de las unidades NAL en el tren de unidades NAL.

El formato tren de bytes se puede obtener a partir del formato tren de unidades NAL para lo cual se ordenan las unidades NAL en el orden de decodificación y se añade como prefijo a cada unidad NAL un código de inicio y quizá algunos bytes de valor cero para formar así un tren de bytes. El formato tren de unidades NAL se puede obtener del formato de tren de bytes, para lo cual se busca la posición del patrón de prefijos código de inicio únicos de ese tren de bytes. Los métodos de entramado de unidades NAL distintos del utilizado en el formato tren de bytes están fuera del alcance de esta Recomendación | Norma Internacional. El formato tren de bytes se especifica en el anexo B.

## 6.2 Formato de imágenes fuente, decodificadas y resultantes

Esta subcláusula especifica la relación entre cuadros y campos fuente y decodificados que vienen dados en el tren de bits.

La fuente de vídeo representada por el tren de bits es una secuencia de cuadros, campos o ambos (denominados genéricamente como imágenes) en orden de decodificación.

Las imágenes (cuadros o campos) fuente y decodificadas están compuestas por tres matrices de muestras, una luma y dos croma.

La variable ChromaFormatFactor se especifica en el cuadro 6-1, en función de la estructura de muestreo del formato croma. El valor de ChromaFormatFactor se supondrá igual a 1,5, lo que indica un muestreo 4:2:0. En el muestreo monocromático sólo hay una matriz de muestras, la cual puede considerarse en teoría una matriz luma. En el muestreo 4:2:0, cada una de las dos matrices croma tiene la mitad de altura y la mitad de anchura de la matriz luma. En el muestreo 4:2:2, cada una de las dos matrices croma tiene la misma altura y la mitad de anchura que la matriz luma. En el muestreo 4:4:4, cada una de las dos matrices croma tiene la misma altura y la misma anchura que la matriz luma.

NOTA – Es posible que haya otros valores válidos en futuras versiones de esta Recomendación | Norma Internacional.

**Cuadro 6-1 – Valores de ChromaFormatFactor**

Formato croma	ChromaFormatFactor
monocromo	1
4:2:0	1,5
4:2:2	2
4:4:4	3

Esta Recomendación | Norma Internacional representa secuencias de color con muestreo croma 4:2:0. La anchura de la matriz de muestras luma de cada imagen es un múltiplo entero de 16. La anchura de las matrices de muestras croma de cada imagen es un múltiplo entero de 8. La altura de la matriz de muestras luma de cada imagen codificada (ya sea un cuadro codificado o un campo codificado) es un múltiplo entero de 16 y la altura de cada matriz croma de esas imágenes es un múltiplo entero de 8. Si hay cuadros codificados en la secuencia de vídeo codificada que contiene campos codificados o contiene cuadros codificados que utilizan la codificación de cuadro/campo adaptativa por macrobloque, la altura de la matriz de muestras luma de todos los cuadros codificados en la secuencia de vídeo codificada es un múltiplo entero de 32 y la altura de cada

matriz de muestras croma de esos cuadros es un múltiplo entero de 16. La anchura o la altura de las imágenes que resultan de aplicar el proceso de decodificación no tienen por qué ser un múltiplo entero de 16 y se puede especificar utilizando un rectángulo de recorte.

La anchura de los campos codificados mediante referencia a un conjunto concreto de parámetros secuencia es la misma que la de los cuadros codificados mediante referencia al mismo conjunto de parámetros secuencia (véase más adelante). La altura de los campos codificados mediante referencia a un conjunto concreto de parámetros secuencia es la mitad que la de los cuadros codificados mediante referencia al mismo conjunto de parámetros secuencia (véase más adelante).

En la figura 6-1 se muestran las posiciones relativas vertical y horizontal nominales de muestras luma y croma en cuadros. Es posible indicar otras posiciones relativas de las muestras croma en la información sobre los posibles usos de vídeo (véase anexo E).



**Figura 6-1 – Posiciones vertical y horizontal nominales de muestras luma y croma 4:2:0 en un cuadro**

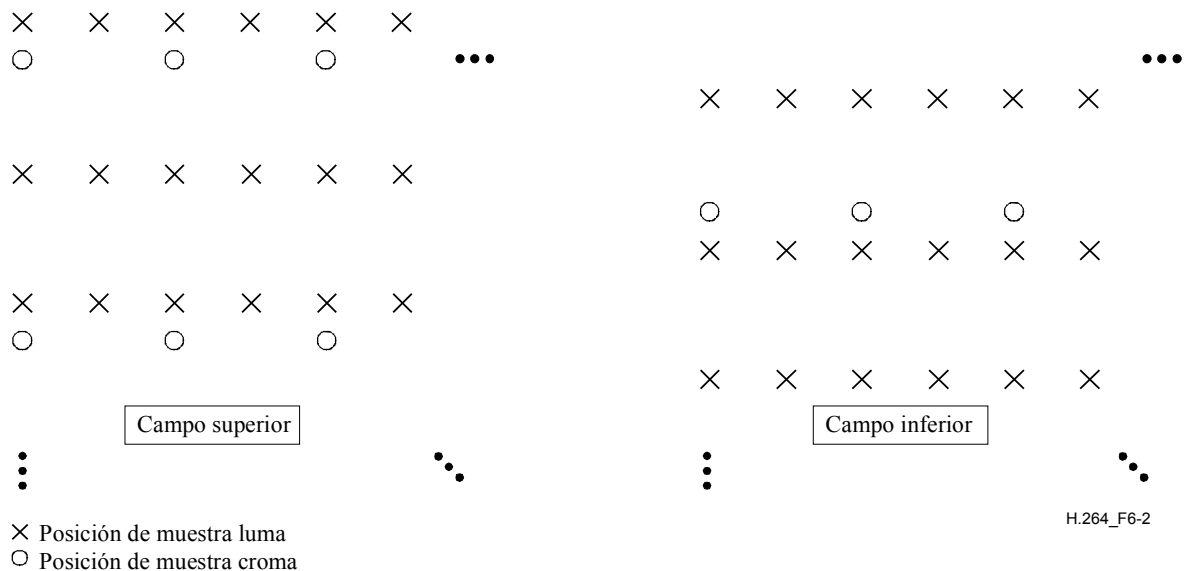
Los cuadros constan de dos campos, como se describe a continuación. Una imagen codificada representa un cuadro codificado o un campo codificado. Las secuencias vídeo codificadas que son conformes con esta Recomendación | Norma Internacional pueden contener cualquier combinación de cuadros codificados y campos codificados. Además, el proceso de decodificación se especifica de modo que se pueda codificar regiones más pequeñas que un cuadro o un campo, para lo cual se utiliza la codificación de cuadro/campo adaptativa por macrobloque.

Los campos fuente y decodificados pueden ser de dos tipos: campo superior o campo inferior. Cuando dos campos se sacan a la vez, o cuando se combinan para utilizarlos como cuadro de referencia (véase más adelante), los dos campos (que serán de paridad contraria) se entrelazan. Las filas primera (es decir la de más arriba), tercera, quinta, etc. de un cuadro decodificado son las filas del campo superior. Las filas segunda, cuarta, sexta, etc. de un cuadro decodificado son las filas del campo inferior. La primera fila (o sea la fila superior) se numera como fila número 0; la segunda fila se numera como fila número 1; la tercera fila se numera como fila número 2, etc. Los campos superiores constan únicamente de las filas del campo superior de un cuadro, y los campos inferiores constan únicamente de las filas del campo inferior de un cuadro. Cuando el campo superior o el campo inferior de un cuadro decodificado se utiliza como campo de referencia (véase más adelante) sólo se utilizan las filas de numeración par (si se trata de un campo superior) o las filas de numeración impar (si se trata de un campo inferior) del cuadro decodificado.

Las posiciones nominales relativas vertical y horizontal de muestras luma y croma en los campos superior e inferior se muestran en la figura 6-2. Las posiciones nominales relativas de muestreo vertical de las muestras croma en el campo superior se especifican como un desplazamiento hacia

arriba de un cuarto de la altura de la muestra luma relativo a la cuadrícula de muestreo del campo. Las posiciones de muestreo vertical de las muestras croma en el campo inferior se especifican como un desplazamiento hacia abajo de un cuarto de la altura de la muestra luma relativo a la cuadrícula de muestreo del campo. Se pueden indicar otras posiciones relativas de las muestras croma en la información sobre los posibles usos de vídeo (véase anexo E).

NOTA – Las muestras croma se desplazan para que estén alineadas verticalmente con la posición usual relativa a la cuadrícula de muestreo de todo el cuadro, como se muestra en la figura 6-1.

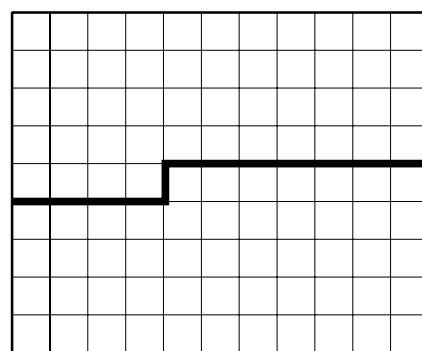


**Figura 6-2 – Posiciones de muestreo vertical y horizontal nominales de los campos de muestra superior e inferior**

### 6.3 Subdivisión espacial de imágenes y sectores

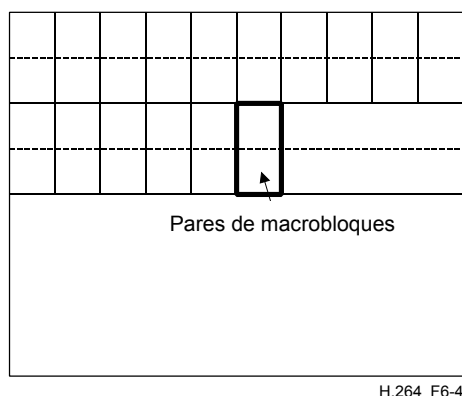
En esta subcláusula se especifica cómo se dividen las imágenes en sectores y macrobloques. Las imágenes se dividen en sectores. Un sector es una secuencia de macrobloques o, cuando se utiliza la decodificación de cuadro/campo adaptativa por macrobloque, una secuencia de pares de macrobloques.

Cada macrobloque está formado por una matriz de muestras luma 16x16 y dos matrices de muestras croma 8x8. Si no se utiliza la decodificación cuadro/campo adaptativa por macrobloque, cada macrobloque representa una región rectangular espacial de la imagen. La figura 6-3 muestra un ejemplo de cómo se puede dividir una imagen en dos sectores.



**Figura 6-3 – Subdivisión de una imagen de 11 por 9 macrobloques en dos sectores**

Cuando se utiliza la decodificación cuadro/campo adaptativa por macrobloque, la imagen se divide en sectores que contiene un número entero de pares de macrobloques, como se muestra en la figura 6-4. Cada par de macrobloques contiene dos macrobloques.



**Figura 6-4 – Subdivisión de un cuadro decodificado en pares de macrobloques**

## 6.4 Procesos de barrido inverso y procesos de obtención de macrobloques adyacentes

Esta subcláusula especifica los procesos de barrido inverso, es decir, el mapeado de índices en posiciones, y los procesos de obtención de macrobloques adyacentes.

### 6.4.1 Proceso de barrido inverso de macrobloques

Este proceso acepta como argumento una dirección de macrobloque, mbAddr.

El proceso genera como resultado la posición ( x, y ) de la muestra luma superior izquierda del macrobloque cuya dirección es mbAddr relativa a la muestra superior izquierda de la imagen.

El proceso de barrido inverso de macrobloques se define del modo siguiente:

- Si MbaffFrameFlag es igual a 0:

$$x = \text{InverseRasterScan}( \text{mbAddr}, 16, 16, \text{PicWidthInSamples}_L, 0 ) \quad (6-1)$$

$$y = \text{InverseRasterScan}( \text{mbAddr}, 16, 16, \text{PicWidthInSamples}_L, 1 ) \quad (6-2)$$

- De lo contrario (MbaffFrameFlag es igual a 1), se aplica lo siguiente:

$$xO = \text{InverseRasterScan}( \text{mbAddr} / 2, 16, 32, \text{PicWidthInSamples}_L, 0 ) \quad (6-3)$$

$$yO = \text{InverseRasterScan}( \text{mbAddr} / 2, 16, 32, \text{PicWidthInSamples}_L, 1 ) \quad (6-4)$$

En función del tipo de macrobloque de que se trate se aplica lo siguiente:

- Si el macrobloque es un macrobloque cuadro:

$$x = xO \quad (6-5)$$

$$y = yO + ( \text{mbAddr} \% 2 ) * 16 \quad (6-6)$$

- De lo contrario (el macrobloque es un macrobloque campo):

$$x = xO \quad (6-7)$$

$$y = yO + ( \text{mbAddr} \% 2 ) \quad (6-8)$$

### 6.4.2 Proceso de barrido inverso de particiones macrobloque y particiones submacrobloque

Los macrobloques y los submacrobloques se pueden subdividirse, y las particiones resultantes se barren para realizar la predicción inter, como se muestra en la figura 6-5. Los rectángulos de contorno indican, respectivamente, las muestras en un macrobloque o submacrobloque. Los rectángulos indican las particiones. El número de cada rectángulo especifica el índice del barrido inverso de particiones macrobloque o del barrido inverso de particiones submacrobloque.

Las funciones MbPartWidth(), MbPartHeight(), SubMbPartWidth() y SubMbPartHeight() describen la anchura y la altura de las particiones macrobloque y de las particiones submacrobloque y se especifican en los cuadros 7-10, 7-11, 7-14 y 7-15. MbPartWidth() y MbPartHeight() se configuran a los valores adecuados para cada macrobloque, en función del tipo de macrobloque. SubMbPartWidth() y SubMbPartHeight() se configuran a los valores adecuados para cada submacrobloque de un macrobloque con mb\_type igual a P\_8x8, P\_8x8ref0, o B\_8x8, en función del tipo de submacrobloque.

	1 partición macrobloque de 16*16 muestras luma y sus correspondientes muestras croma	2 particiones macrobloque de 16*8 muestras luma y sus correspondientes muestras croma	2 particiones macrobloque de 8*16 muestras luma y sus correspondientes muestras croma	4 particiones submacrobloque de 8*8 muestras luma y sus correspondientes muestras croma
<b>Particiones macrobloque</b>				
	1 partición submacrobloque de 8*8 muestras luma y sus muestras croma asociadas	2 particiones submacrobloque de 8*4 muestras luma y sus muestras croma asociadas	2 particiones submacrobloque de 4*8 muestras luma y sus muestras croma asociadas	4 particiones submacrobloque de 4*4 muestras luma y sus muestras croma asociadas
<b>Particiones submacrobloque</b>				

H.264\_F6-5

**Figura 6-5 – Particiones macrobloque, particiones submacrobloque, barrido de particiones macrobloque y barrido de particiones submacrobloque**

#### 6.4.2.1 Proceso de barrido inverso de particiones de macrobloque

Este proceso acepta como argumento el índice de una partición macrobloque, mbPartIdx.

Este proceso genera como resultado la posición ( x, y ) de la muestra luma superior izquierda de la partición macrobloque mbPartIdx relativa a la muestra superior izquierda del macrobloque.

El proceso de barrido inverso de particiones macrobloque se especifica así:

$$x = \text{InverseRasterScan}( \text{mbPartIdx}, \text{MbPartWidth}( \text{mb\_type} ), \text{MbPartHeight}( \text{mb\_type} ), 16, 0 ) \quad (6-9)$$

$$y = \text{InverseRasterScan}( \text{mbPartIdx}, \text{MbPartWidth}( \text{mb\_type} ), \text{MbPartHeight}( \text{mb\_type} ), 16, 1 ) \quad (6-10)$$

#### 6.4.2.2 Proceso de barrido inverso de particiones submacrobloque

Este proceso acepta como argumento el índice de una partición macrobloque, mbPartIdx, y el índice de una partición submacrobloque, subMbPartIdx.

Este proceso genera como resultado la posición ( x, y ) de la muestra luma superior izquierda de la partición submacrobloque, subMbPartIdx, relativa a la muestra superior izquierda del submacrobloque.

El proceso de barrido inverso de particiones submacrobloque se define del modo siguiente:

– Si mb\_type es igual a P\_8x8, P\_8x8ref0, o B\_8x8:

$$x = \text{InverseRasterScan}( \text{subMbPartIdx}, \text{SubMbPartWidth}( \text{sub\_mb\_type}[ \text{mbPartIdx} ] ), \text{SubMbPartHeight}( \text{sub\_mb\_type}[ \text{mbPartIdx} ] ), 8, 0 ) \quad (6-11)$$

$$y = \text{InverseRasterScan}( \text{subMbPartIdx}, \text{SubMbPartWidth}( \text{sub\_mb\_type}[ \text{mbPartIdx} ] ), \text{SubMbPartHeight}( \text{sub\_mb\_type}[ \text{mbPartIdx} ] ), 8, 1 ) \quad (6-12)$$

– De lo contrario,

$$x = \text{InverseRasterScan}(\text{ subMbPartIdx}, 4, 4, 8, 0) \quad (6-13)$$

$$y = \text{InverseRasterScan}(\text{ subMbPartIdx}, 4, 4, 8, 1) \quad (6-14)$$

### 6.4.3 Proceso de barrido inverso de bloques luma 4x4

Este proceso acepta como argumento el índice de un bloque luma 4x4, luma4x4BlkIdx.

Este proceso genera como resultado la posición ( x, y ) de la muestra luma superior izquierda del bloque luma 4x4, con índice luma4x4BlkIdx, relativo a la muestra luma superior izquierda del macrobloque.

La figura 6-6 muestra el barrido de bloques luma 4x4.

0	1	4	5
2	3	6	7
8	9	12	13
10	11	14	15

**Figura 6-6 – Barrido de bloques luma 4x4**

El proceso de barrido inverso de bloques luma 4x4 se especifica mediante:

$$x = \text{InverseRasterScan}(\text{ luma4x4BlkIdx} / 4, 8, 8, 16, 0) + \text{InverseRasterScan}(\text{ luma4x4BlkIdx} \% 4, 4, 4, 8, 0) \quad (6-15)$$

$$y = \text{InverseRasterScan}(\text{ luma4x4BlkIdx} / 4, 8, 8, 16, 1) + \text{InverseRasterScan}(\text{ luma4x4BlkIdx} \% 4, 4, 4, 8, 1) \quad (6-16)$$

### 6.4.4 Proceso de obtención de la disponibilidad de direcciones de macrobloque

Este proceso acepta como argumento una dirección de macrobloque, mbAddr.

Este proceso genera como resultado la disponibilidad del macrobloque mbAddr.

NOTA – El significado de disponibilidad se determina al llamar a este proceso.

El macrobloque se marca como disponible, salvo si se cumple alguna de las siguientes condiciones, en cuyo caso el macrobloque se marca como no disponible:

- mbAddr < 0
- mbAddr > CurrMbAddr
- el macrobloque con dirección mbAddr está en un sector diferente del sector considerado.

### 6.4.5 Proceso de obtención de direcciones de macrobloques adyacentes y su disponibilidad

Sólo se puede llamar a este proceso cuando MbaffFrameFlag es igual a 0.

Este proceso genera como resultado:

- mbAddrA: dirección y estado de disponibilidad del macrobloque situado a la izquierda del macrobloque considerado.
- mbAddrB: dirección y estado de disponibilidad del macrobloque situado encima del macrobloque considerado.
- mbAddrC: dirección y estado de disponibilidad del macrobloque situado arriba y a la derecha del macrobloque considerado.
- mbAddrD: dirección y estado de disponibilidad del macrobloque situado arriba y a la izquierda del macrobloque considerado.

En la figura 6-7 se muestra las posiciones espaciales relativas de los macrobloques correspondientes a mbAddrA, mbAddrB, mbAddrC, y mbAddrD relativas al macrobloque considerado, CurrMbAddr.

mbAddrD	mbAddrB	mbAddrC
mbAddrA	CurrMbAddr r	

**Figura 6-7 – Macrobloques adyacentes a un determinado macrobloque**

El proceso descrito en la subcláusula 6.4.4 acepta como argumento  $mbAddrA = CurrMbAddr - 1$  y genera como resultado la disponibilidad del macrobloque mbAddrA. Además, mbAddrA se marca como no disponible cuando  $CurrMbAddr \% PicWidthInMbs$  es igual a 0.

El proceso descrito en la subcláusula 6.4.4 acepta como argumento  $mbAddrB = CurrMbAddr - PicWidthInMbs$  y genera como resultado la disponibilidad del macrobloque mbAddrB.

El proceso descrito en la subcláusula 6.4.4 acepta como argumento  $mbAddrC = CurrMbAddr - PicWidthInMbs + 1$  y genera como resultado la disponibilidad del macrobloque mbAddrC. Además, mbAddrC se marca como no disponible cuando  $(CurrMbAddr + 1) \% PicWidthInMbs$  es igual a 0.

El proceso descrito en la subcláusula 6.4.4 acepta como argumento  $mbAddrD = CurrMbAddr - PicWidthInMbs - 1$  y genera como resultado la disponibilidad del macrobloque mbAddrD. Además, mbAddrD se marca como no disponible cuando  $CurrMbAddr \% PicWidthInMbs$  es igual a 0.

#### **6.4.6 Proceso de obtención de direcciones de macrobloques adyacentes y su disponibilidad en cuadros MBAFF**

Sólo se puede llamar a este proceso cuando MbaffFrameFlag es igual a 1.

Este proceso genera como resultado:

- mbAddrA: dirección y estado de disponibilidad del macrobloque superior del par de macrobloques situado a la izquierda del par de macrobloques considerado.
- mbAddrB: dirección y estado de disponibilidad del macrobloque superior del par de macrobloques situado arriba del par de macrobloques considerado.
- mbAddrC: dirección y estado de disponibilidad del macrobloque superior del par de macrobloques situado arriba y a la derecha del par de macrobloques considerado.
- mbAddrD: dirección y estado de disponibilidad del macrobloque superior del par de macrobloques situado arriba y a la izquierda del par de macrobloques considerado.

La figura 6-8 muestra las posiciones espaciales relativas de los macrobloques con mbAddrA, mbAddrB, mbAddrC, y mbAddrD relativas al macrobloque considerado, CurrMbAddr.

Los valores de mbAddrA, mbAddrB, mbAddrC, y mbAddrD son iguales independientemente de si el macrobloque considerado es el superior o el inferior de un par de macrobloques.



mbAddrD	mbAddrB	mbAddrC
mbAddrA	CurrMbAddr o	
	CurrMbAddr	

**Figura 6-8 – Macrobloques adyacentes de un determinado macrobloque en cuadros MBAFF**

El proceso descrito en la subcláusula 6.4.4 acepta como argumento  $mbAddrA = 2 * ( CurrMbAddr / 2 - 1 )$  y genera como resultado la disponibilidad del macrobloque mbAddrA. Además, mbAddrA se marca como no disponible cuando  $( CurrMbAddr / 2 ) \% PicWidthInMbs$  es igual a 0.

El proceso descrito en la subcláusula 6.4.4 acepta como argumento  $mbAddrB = 2 * ( CurrMbAddr / 2 - PicWidthInMbs )$  y genera como resultado la disponibilidad del macrobloque mbAddrB.

El proceso descrito en la subcláusula 6.4.4 acepta como argumento  $mbAddrC = 2 * ( CurrMbAddr / 2 - PicWidthInMbs + 1 )$  y genera como resultado la disponibilidad del macrobloque mbAddrC. Además, mbAddrC se marca como no disponible cuando  $( CurrMbAddr / 2 + 1 ) \% PicWidthInMbs$  es igual a 0.

El proceso descrito en la subcláusula 6.4.4 acepta como argumento  $mbAddrD = 2 * ( CurrMbAddr / 2 - PicWidthInMbs - 1 )$  y genera como resultado la disponibilidad del macrobloque mbAddrD. Además, mbAddrD se marca como no disponible cuando  $( CurrMbAddr / 2 ) \% PicWidthInMbs$  es igual a 0.

#### **6.4.7 Procesos de obtención de macrobloques, bloques y particiones adyacentes**

La subcláusula 6.4.7.1 especifica el proceso de obtención de macrobloques adyacentes.

La subcláusula 6.4.7.2 especifica el proceso de obtención de bloques luma 8x8 adyacentes.

La subcláusula 6.4.7.3 especifica el proceso de obtención de bloques luma 4x4 adyacentes.

La subcláusula 6.4.7.4 especifica el proceso de obtención de bloques croma 4x4 adyacentes.

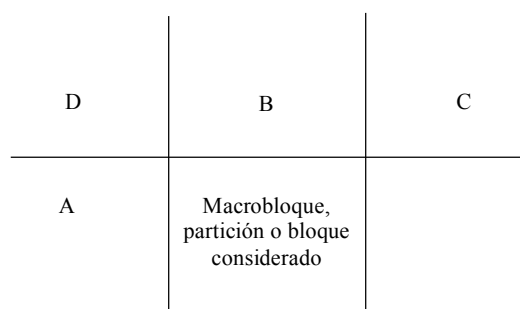
La subcláusula 6.4.7.5 especifica el proceso de obtención de particiones adyacentes.

En el cuadro 6-2 se especifican los valores de la diferencia de posición luma (xD, yD) correspondientes al argumento y el valor de N en mbAddrN, mbPartIdxN, subMbPartIdxN, luma8x8BlkIdxN, luma4x4BlkIdxN, y chroma4x4BlkIdxN correspondientes al resultado. Estas asignaciones de argumento y resultado se utilizan en las subcláusulas 6.4.7.1 a 6.4.7.5. La variable predPartWidth se especifica cuando se hace referencia al cuadro 6-2.

**Cuadro 6-2 – Especificación de las asignaciones de argumento y resultado que se utilizan en las subcláusulas 6.4.7.1 a 6.4.7.5**

N	xD	yD
A	-1	0
B	0	-1
C	predPartWidth	-1
D	-1	-1

La figura 6-9 ilustra la posición relativa de los macrobloques, bloques o particiones A, B, C, y D adyacentes al macrobloque, partición, o bloque considerado, cuando el macrobloque, partición o bloque considerado está en modo codificación cuadro.



H.264\_F6-9

**Figura 6-9 – Determinación de macrobloques, bloques y particiones adyacentes (informativo)**

#### 6.4.7.1 Proceso de obtención de macrobloques adyacentes

Este proceso genera como resultado:

- mbAddrA: la dirección del macrobloque situado a la izquierda del macrobloque considerado y su estado de disponibilidad, y
- mbAddrB: dirección del macrobloque situado encima del macrobloque considerado y su estado de disponibilidad.

mbAddrN (siendo N = A o B) se obtiene del modo siguiente:

- La diferencia de posición luma ( xD, yD ) se fija de acuerdo con el cuadro 6-2.
- Se llama al proceso de obtención de posiciones adyacentes especificado en la subcláusula 6.4.8 para las posiciones luma ( xN, yN ) igual a ( xD, yD ), y el resultado se asigna a mbAddrN.

#### 6.4.7.2 Proceso de obtención de bloques luma 8x8 adyacentes

Este proceso acepta como argumento un índice de bloque luma 8x8, luma8x8BlkIdx.

El luma8x8BlkIdx especifica los bloques luma 8x8 de un macrobloque en un barrido por filas.

Este proceso genera como resultado:

- mbAddrA: cuyo valor es CurrMbAddr o la dirección del macrobloque situado a la izquierda del macrobloque considerado y su estado de disponibilidad;
- luma8x8BlkIdxA: índice del bloque luma 8x8 situado a la izquierda del bloque 8x8 con índice luma8x8BlkIdx y su estado de disponibilidad;

- mbAddrB: cuyo valor es CurrMbAddr o la dirección del macrobloque situado encima del macrobloque considerado y su estado de disponibilidad;
- luma8x8BlkIdxB: índice del bloque luma 8x8 situado encima del bloque 8x8 con índice luma8x8BlkIdx y su estado de disponibilidad.

mbAddrN y luma8x8BlkIdxN (siendo N = A o B) se obtienen del modo siguiente:

- La diferencia de posición luma ( xD, yD ) se fija de acuerdo con el cuadro 6-2.

- La posición luma ( xN, yN ) se especifica mediante:

$$xN = ( \text{luma8x8BlkIdx} \% 2 ) * 8 + xD \quad (6-17)$$

$$yN = ( \text{luma8x8BlkIdx} / 2 ) * 8 + yD \quad (6-18)$$

- Se llama al proceso de obtención de posiciones adyacentes descrito en la subcláusula 6.4.8, se le pasan como argumentos las posiciones luma ( xN, yN ) y el resultado se asigna a mbAddrN y ( xW, yW ).
- La variable luma8x8BlkIdxN se obtiene del siguiente modo:
  - Si mbAddrN no está disponible, luma8x8BlkIdxN se marca como no disponible.
  - De lo contrario (mbAddrN está disponible), se asigna a luma8x8BlkIdxN el bloque luma 8x8 en el macrobloque mbAddrN que abarca la posición luma ( xW, yW ).

### 6.4.7.3 Proceso de obtención de bloques luma 4x4 adyacentes

Este proceso acepta como argumento un índice de bloque luma 4x4, luma4x4BlkIdx.

Este proceso genera como resultado:

- mbAddrA: cuyo valor es CurrMbAddr o la dirección del macrobloque situado a la izquierda del macrobloque considerado y su estado de disponibilidad;
- luma4x4BlkIdxA: índice del bloque luma 4x4 situado a la izquierda del bloque 4x4 con índice luma4x4BlkIdx y su estado de disponibilidad;
- mbAddrB: cuyo valor es CurrMbAddr o la dirección del macrobloque situado encima del macrobloque considerado y su estado de disponibilidad;
- luma4x4BlkIdxB: índice del bloque luma 4x4 situado encima del bloque 4x4 con índice luma4x4BlkIdx y su estado de disponibilidad.

mbAddrN y luma4x4BlkIdxN (siendo N = A o B) se obtienen del modo siguiente:

- La diferencia de posición luma ( xD, yD ) se fija de acuerdo con el cuadro 6-2.

- Se llama al proceso de barrido inverso de bloques luma 4x4 descrito en la subcláusula 6.4.3, se le pasa como argumento luma4x4BlkIdx y el resultado es ( x, y ).

- La posición luma ( xN, yN ) se especifica mediante:

$$xN = x + xD \quad (6-19)$$

$$yN = y + yD \quad (6-20)$$

- Se llama al proceso de obtención de posiciones adyacente descrito en la subcláusula 6.4.8, pasándole como argumentos las posiciones luma ( xN, yN ) y el resultado se asigna a mbAddrN y ( xW, yW ).
- La variable luma4x4BlkIdxN se obtiene del siguiente modo:
  - Si mbAddrN no está disponible, se marca luma4x4BlkIdxN como no disponible.
  - De lo contrario (mbAddrN está disponible), se asigna a luma4x4BlkIdxN el bloque luma 4x4 en el macrobloque mbAddrN que abarca la posición luma ( xW, yW ).

#### 6.4.7.4 Proceso de obtención de bloques croma 4x4 adyacentes

Este proceso acepta como argumento un bloque croma 4x4, chroma4x4BlkIdx.

Este proceso genera como resultado:

- mbAddrA: cuyo valor es CurrMbAddr o la dirección del macrobloque situado a la izquierda del macrobloque considerado y su estado de disponibilidad;
- chroma4x4BlkIdxA: el índice del bloque croma 4x4 situado a la izquierda del bloque croma 4x4 con índice chroma4x4BlkIdx y su estado de disponibilidad;
- mbAddrB: es igual a CurrMbAddr o la dirección del macrobloque situado encima del macrobloque actual y su estado de disponibilidad;
- chroma4x4BlkIdxB: el índice del bloque croma 4x4 situado encima del bloque croma 4x4 con índice chroma4x4BlkIdx y su estado de disponibilidad.

Se llama al proceso de obtención de bloques luma 8x8 adyacentes, pasándole como argumento luma8x8BlkIdx = chroma4x4BlkIdx y el resultado es mbAddrA, chroma4x4BlkIdxA = luma8x8BlkIdxA, mbAddrB, y chroma4x4BlkIdxB = luma8x8BlkIdxB.

#### 6.4.7.5 Proceso de obtención de particiones adyacentes

Este proceso acepta como argumentos:

- un índice de partición macrobloque, mbPartIdx;
- un tipo de submacrobloque considerado, currSubMbType;
- un índice de partición submacrobloque, subMbPartIdx.

El proceso genera como resultado:

- mbAddrA\mbPartIdxA\subMbPartIdxA: especifica la partición macrobloque o submacrobloque situada a la izquierda del macrobloque considerado y su estado de disponibilidad, o la partición submacrobloque CurrMbAddr\mbPartIdx\subMbPartIdx y su estado de disponibilidad;
- mbAddrB\mbPartIdxB\subMbPartIdxB: especifica la partición macrobloque o submacrobloque situada encima del macrobloque considerado y su estado de disponibilidad, o la partición submacrobloque CurrMbAddr\mbPartIdx\subMbPartIdx y su estado de disponibilidad;
- mbAddrC\mbPartIdxC\subMbPartIdxC: especifica la partición macrobloque o submacrobloque situada arriba y a la derecha del macrobloque considerado y su estado de disponibilidad, o la partición submacrobloque CurrMbAddr\mbPartIdx\subMbPartIdx y su estado de disponibilidad;
- mbAddrD\mbPartIdxD\subMbPartIdxD: especifica la partición macrobloque o submacrobloque situada encima y a la izquierda del macrobloque considerado y su estado de disponibilidad, o la partición submacrobloque CurrMbAddr\mbPartIdx\subMbPartIdx y su estado de disponibilidad.

mbAddrN, mbPartIdxN, y subMbPartIdx (siendo N = A, B, C, o D) se obtienen del siguiente modo:

- Se llama al proceso de barrido inverso de particiones macrobloque descrito en la subcláusula 6.4.2.1, se le pasa como argumento mbPartIdx y el resultado es ( x, y ).
- La posición de la muestra luma superior izquierda dentro de la partición macrobloque ( xS, yS ) se obtiene del siguiente modo:
  - Si mb\_type es igual a P\_8x8, P\_8x8ref0 o B\_8x8, se llama al proceso de barrido inverso de partición de submacrobloque descrito en la subcláusula 6.4.2.2, pasándole como argumento subMbPartIdx y el resultado es ( xS, yS ).
  - De lo contrario, ( xS, yS ) se fijan a ( 0, 0 ).

- La variable `predPartWidth` del cuadro 6-2 se especifica del siguiente modo:
    - Si `mb_type` es igual a `P_Skip`, `B_Skip`, o `B_Direct_16x16`, `predPartWidth = 16`.
    - De lo contrario, si `mb_type` es igual a `B_8x8`, se aplica lo siguiente.
      - Si `currSubMbType` es igual a `B_Direct_8x8`, `predPartWidth = 16`.  
 NOTA – Cuando `currSubMbType` es igual a `B_Direct_8x8` y `direct_spatial_mv_pred_flag` es igual a 1, el vector de movimiento predicho es el correspondiente al macrobloque completo.
      - De lo contrario, `predPartWidth = SubMbPartWidth( sub_mb_type[ mbPartIdx ] )`.
    - De lo contrario, si `mb_type` es igual a `P_8x8`, o `P_8x8ref0`, `predPartWidth = SubMbPartWidth( sub_mb_type[ mbPartIdx ] )`.
    - De lo contrario, `predPartWidth = MbPartWidth( mb_type )`.
  - La diferencia de posición luma ( `xD`, `yD` ) se fija de conformidad con el cuadro 6-2.
  - La posición luma adyacente ( `xN`, `yN` ) viene dada por:
 
$$xN = x + xS + xD \quad (6-21)$$

$$yN = y + yS + yD \quad (6-22)$$
  - Se llama al proceso de obtención de posiciones adyacentes descrito en las subcláusula 6.4.8, se le pasa como argumentos las posiciones luma ( `xN`, `yN` ) y el resultado se asigna a `mbAddrN` y ( `xW`, `yW` ).
  - En función de `mbAddrN`, se aplica lo siguiente:
    - Si `mbAddrN` no está disponible, la partición macrobloque o submacrobloque `mbAddrN\mbPartIdxN\subMbPartIdxN` se marca como no disponible.
    - De lo contrario (`mbAddrN` está disponible), se aplica lo siguiente:
      - Se asigna a `mbPartIdxN` la partición macrobloque en el macrobloque `mbAddrN` que abarca la posición luma ( `xW`, `yW` ) y se asigna a `subMbPartIdxN` la partición submacrobloque dentro de la partición macrobloque `mbPartIdxN` que abarca la muestra ( `xW`, `yW` ) en el macrobloque `mbAddrN`.
      - Si la partición dada por `mbPartIdxN` y `subMbPartIdxN` todavía no está decodificada, la partición macrobloque `mbPartIdxN` y la partición submacrobloque `subMbPartIdxN` se marcan como no disponibles.
- NOTA – La última condición se produce, cuando, por ejemplo, `mbPartIdx = 2`, `subMbPartIdx = 3`, `xD = 4`, `yD = -1`, es decir, cuando se solicita el C adyacente del último bloque luma de 4x4 del tercer submacrobloque.

#### 6.4.8 Proceso de obtención de posiciones adyacentes

Este proceso acepta como argumento una posición luma o croma ( `xN`, `yN` ) relativa a la esquina superior izquierda del macrobloque considerado.

El proceso genera como resultado:

- `mbAddrN`: cuyo valor es `CurrMbAddr` o la dirección del macrobloque adyacente que contiene ( `xN`, `yN` ) y su estado de disponibilidad;
- ( `xW`, `yW` ): la posición ( `xN`, `yN` ) relativa a la esquina superior izquierda del macrobloque `mbAddrN` (en lugar de la relativa a la esquina superior izquierda del macrobloque considerado).

Sea `maxWH` una variable que especifica el valor máximo de la posición de las componentes `xN`, `yN`, `xW`, e `yW`. El valor de `maxWH` se obtiene del modo siguiente:

- Si se aplica este proceso a posiciones luma adyacentes:
 
$$\text{maxWH} = 16 \quad (6-23)$$

- En caso contrario (se aplica este proceso a posiciones cromas adyacentes):

$$\text{maxWH} = 8 \quad (6-24)$$

En función de la variable MbaffFrameFlag, las posiciones luma adyacentes se calculan del modo siguiente:

- Si MbaffFrameFlag es igual a 0, se aplica la especificación descrita en la subcláusula 6.4.8.1 para las posiciones luma adyacentes en campos y cuadros distintos de los MBAFF.
- De lo contrario (MbaffFrameFlag es igual a 1), se aplica la especificación descrita en la subcláusula 6.4.8.2 para las posiciones luma adyacentes en los cuadros MBAFF.

#### 6.4.8.1 Especificación de posiciones luma adyacentes en campos y cuadros distintos de los MBAFF

Las especificaciones de esta subcláusula se aplican cuando MbaffFrameFlag es igual a 0.

Se llama al proceso de obtención de direcciones de macrobloques adyacentes y su disponibilidad descrito en la subcláusula 6.4.5, pasándole como argumentos mbAddrA, mbAddrB, mbAddrC, y mbAddrD y el resultado es el estado de disponibilidad.

El cuadro 6-3 especifica mbAddrN en función de ( xN, yN ).

**Cuadro 6-3 – Especificación de mbAddrN**

xN	yN	mbAddrN
< 0	< 0	mbAddrD
< 0	0 .. maxWH – 1	mbAddrA
0 .. maxWH – 1	< 0	mbAddrB
0 .. maxWH – 1	0 .. maxWH – 1	CurrMbAddr
> maxWH – 1	< 0	mbAddrC
> maxWH – 1	0 .. maxWH – 1	no disponible
	> maxWH – 1	no disponible

La posición luma adyacente ( xW, yW ) relativa a la esquina superior izquierda del macrobloque mbAddrN se calcula del modo siguiente:

$$xW = ( xN + \text{maxWH} ) \% \text{maxWH} \quad (6-25)$$

$$yW = ( yN + \text{maxWH} ) \% \text{maxWH} \quad (6-26)$$

#### 6.4.8.2 Especificación de posiciones luma adyacentes en cuadros MBAFF

Las especificaciones de esta subcláusula se aplican cuando MbaffFrameFlag es igual a 1.

Se llama al proceso de obtención de direcciones de macrobloques adyacentes y su disponibilidad descrito en la subcláusula 6.4.6, pasándole como argumentos mbAddrA, mbAddrB, mbAddrC, y mbAddrD, y se obtiene como resultado el estado de disponibilidad.

El cuadro 6-4 especifica las direcciones de macrobloques mbAddrN e yM en dos pasos:

1. Especificación de la dirección del macrobloque mbAddrX en función de ( xN, yN ) y las siguientes variables:
  - La variable currMbFrameFlag se calcula del siguiente modo:
    - si el macrobloque con la dirección CurrMbAddr es un macrobloque cuadro, currMbFrameFlag se pone a 1;

- de lo contrario (el macrobloque con la dirección CurrMbAddr es un macrobloque campo), currMbFrameFlag se pone a 0.
  - La variable mbIsTopMbFlag se calcula del modo siguiente:
    - si el macrobloque con la dirección CurrMbAddr es un macrobloque superior (CurrMbAddr % 2 es igual a 0), mbIsTopMbFlag se pone a 1;
    - de lo contrario (el macrobloque con dirección CurrMbAddr es un macrobloque inferior, CurrMbAddr % 2 es igual a 1), mbIsTopMbFlag se pone a 0.
2. En función de la disponibilidad de mbAddrX, se aplica lo siguiente:
- Si mbAddrX no está disponible, mbAddrN se marca como no disponible.
  - De lo contrario (mbAddrX está disponible), mbAddrN se marca como disponible y el cuadro 6-4 especifica mbAddrN e yM en función de (xN, yN), currMbFrameFlag, mbIsTopMbFlag, y la variable mbAddrXFrameFlag, la cual se calcula del modo siguiente:
    - si el macrobloque mbAddrX es un macrobloque cuadro, mbAddrXFrameFlag se pone a 1;
    - de lo contrario (el macrobloque mbAddrX es un macrobloque campo), mbAddrXFrameFlag se pone a 0.

Cuando el valor asignado en el cuadro 6-4 a los indicadores anteriores no está especificado (na) significa que el valor del correspondiente indicador no es pertinente para las filas del cuadro considerado.

**Cuadro 6-4 – Especificación de mbAddrN e yM**

<b>xN</b>	<b>yN</b>	<b>currMbFrameFlag</b>	<b>mbIsTopMbFlag</b>	<b>mbAddrX</b>	<b>mbAddrXFrameFlag</b>	<b>condición adicional</b>	<b>mbAddrN</b>	<b>yM</b>	
< 0	< 0	1	1	mbAddrD			mbAddrD + 1	yN	
			0	mbAddrA	1		mbAddrA	yN	
		0	1	mbAddrD	0		mbAddrA + 1	(yN + maxWH) >> 1	
			0	mbAddrD	1		mbAddrD + 1	2*yN	
< 0	0 .. maxWH - 1	1	1	mbAddrA	1		mbAddrA	yN	
					0	yN% 2 == 0	mbAddrA	yN >> 1	
					0	yN% 2 != 0	mbAddrA + 1	yN >> 1	
			0	mbAddrA	1		mbAddrA + 1	yN	
					0	yN% 2 == 0	mbAddrA	(yN + maxWH) >> 1	
					0	yN% 2 != 0	mbAddrA + 1	(yN + maxWH) >> 1	
		0	mbAddrA	1	1	yN < (maxWH / 2)	mbAddrA	yN << 1	
					1	yN ≥ (maxWH / 2)	mbAddrA + 1	(yN << 1) - maxWH	
					0		mbAddrA	yN	
				0	mbAddrA	1	yN < (maxWH / 2)	mbAddrA	(yN << 1) + 1
						1	yN ≥ (maxWH / 2)	mbAddrA + 1	(yN << 1) + 1 - maxWH
						0		mbAddrA + 1	yN
0 .. maxWH - 1	< 0	1	1	mbAddrB			mbAddrB + 1	yN	
			0	CurrMbAddr			CurrMbAddr - 1	yN	
		0	1	mbAddrB	1		mbAddrB + 1	2 * yN	
			0	mbAddrB	0		mbAddrB	yN	
0 .. maxWH - 1	0 .. maxWH - 1			CurrMbAddr			CurrMbAddr	yN	
> maxWH - 1	< 0	1	1	mbAddrC			mbAddrC + 1	yN	
			0	no disponible			no disponible	na	
		0	1	mbAddrC	1		mbAddrC + 1	2 * yN	
			0	mbAddrC	0		mbAddrC	yN	
> maxWH - 1	0 .. maxWH - 1			no disponible			no disponible	na	
	> maxWH - 1			no disponible			no disponible	na	



La posición luma adyacente (  $x_W$ ,  $y_W$  ) relativa a la esquina superior izquierda del macrobloque  $mbAddr_N$  se calcula del modo siguiente:

$$x_W = ( x_N + \max_{WH} ) \% \max_{WH} \quad (6-27)$$

$$y_W = ( y_M + \max_{WH} ) \% \max_{WH} \quad (6-28)$$

## 7 Sintaxis y semántica

### 7.1 Métodos de describir la sintaxis en forma tabular

Los cuadros de sintaxis describen un superconjunto de la sintaxis de todos los trenes de bits aceptables. Se pueden especificar otras restricciones adicionales de la sintaxis en otras cláusulas.

NOTA – Los decodificadores reales deben implementar mecanismos para identificar los puntos de entrada en el tren de bits para identificar y manipular trenes de bits no conformes. No se describen los métodos de determinación y tratamiento de errores y otras situaciones de este tipo.

El siguiente cuadro muestra ejemplos de pseudocódigo utilizado para describir la sintaxis. Cuando aparece `syntax_element` (elemento sintáctico), éste especifica que el elemento de datos se lee (obtiene) del tren de bits y del puntero al tren de bits.

	<b>C</b>	<b>Descriptor</b>
/* Una expresión puede ser un elemento sintáctico con una categoría sintáctica asociada y un descriptor o puede ser una expresión que especifique condiciones de existencia, de tipo y cuantitativas de elementos sintácticos, como los dos ejemplos siguientes */		
<b>syntax_element</b>	3	ue(v)
expresión condicional		
/* Un grupo de expresiones encerradas entre llaves es una expresión compuesta y funcionalmente se trata como si fuera una expresión sencilla */		
{		
expresión		
expresión		
...		
}		
/* La estructura "while" especifica la comprobación de una condición, y si ésta es verdadera especifica la expresión (o la expresión compuesta) que se repite hasta que la condición deja de ser verdadera */		
while( condición )		
expresión		
/* La estructura "do ... while" especifica una expresión que se aplica una vez, seguido de una comprobación de una condición, y si ésta es verdadera se repite la expresión anterior hasta que la condición deja de ser verdadera */		
do		
expresión		
while( condición )		
/* La estructura "if ... else" especifica la comprobación de una condición, y si ésta es verdadera especifica una expresión primaria, de lo contrario especifica una expresión alternativa. La parte "else" de la estructura y la expresión alternativa se omiten en caso de que ésta no sea necesaria */		
if( condición )		
expresión primaria		
else		
expresión alternativa		
/* La estructura "for" especifica una expresión inicial, seguida de la comprobación de una condición que, si es verdadera, especifica una expresión primaria que se repite seguida de una expresión subsiguiente hasta que la condición deja de ser verdadera */		
for( expresión inicial; condición; expresión subsiguiente )		
expresión primaria		

## 7.2 Especificación de la sintaxis de funciones, categorías y descriptores

En esta subcláusula se describen las funciones que se utilizan en las descripciones sintácticas. En las funciones se supone que existe un puntero al tren de bits que indica la posición del siguiente bit que leerá del tren de bits el proceso de codificación.

`byte_aligned()` se define del modo siguiente:

- Si la posición actual en el tren de bits es un límite de byte, es decir, el siguiente bit en el tren de bits es el primer bit de un byte, el valor que devuelve `byte_aligned()` es TRUE (VERDADERO).
- De lo contrario, el valor que devuelve `byte_aligned()` es FALSE (FALSO).

`more_data_in_byte_stream()` se utiliza únicamente en la estructura sintáctica unidad NAL del tren de bytes especificada en el anexo B, y se define del modo siguiente:

- Si hay más datos en el tren de bytes, el valor que retorna `more_data_in_byte_stream()` es TRUE.
- De lo contrario, el valor que devuelve a `more_data_in_byte_stream()` es FALSE.

`more_rbsp_data()` se define del modo siguiente:

- Si hay más datos en la RBSP antes de `rbsp_trailing_bits()`, el valor que devuelve `more_rbsp_data()` es TRUE.
- De lo contrario, el valor que devuelve `more_rbsp_data()` es FALSE.

El método para habilitar la determinación de si hay más datos en la RBSP depende de la aplicación (el anexo B se describe para aplicaciones que utilizan el formato de tren de bytes).

`more_rbsp_trailing_data()` se define del modo siguiente:

- Si hay más datos en la RBSP, el valor que devuelve `more_rbsp_trailing_data()` es TRUE.
- De lo contrario el valor que devuelve `more_rbsp_trailing_data()` es FALSE.

`next_bits(n)` lee los siguientes bits en el flujo de bits con objeto de hacer comparaciones, sin avanzar el puntero al tren de bits. Se utiliza para leer los siguientes `n` bits del tren de bits, siendo `n` su argumento. Cuando se utiliza en un tren de bytes como el especificado en el anexo B, `next_bits(n)` devuelve 0 si quedan menos de `n` bits en el tren de bytes.

`read_bits(n)` lee los siguientes `n` bits del tren de bits y avanza el puntero al tren de bits `n` posiciones de bit. Cuando `n` es igual a 0, `read_bits(n)` devuelve 0 y no avanza el puntero al tren de bits.

Las categorías (columna C del cuadro) especifican la partición de los datos de sector en tres particiones de datos de sectores como máximo. La partición A de datos de sector contiene todos los elementos sintácticos de la categoría 2. La partición B de datos de sector contiene todos los elementos sintácticos de la categoría 3. La partición C de datos de sector contiene todos los elementos sintácticos de la categoría 4. El significado de los valores de otras categorías no está especificado. En algunos elementos sintácticos se indican dos categorías, separadas por una barra vertical. En esos casos el valor de la categoría que se aplicará se especifica en el texto. En las estructuras sintácticas dentro de otras estructuras sintácticas, se indican las categorías de todos los elementos sintácticos de la estructura sintáctica anidada, separadas por una barra vertical. Los elementos sintácticos o estructuras sintácticas cuya categoría sea "Todas" aparecen dentro de todas las estructuras sintácticas que incluyen ese elemento sintáctico o estructura sintáctica. En las estructuras sintácticas que se utilizan dentro de otras estructuras sintácticas, se considera que se aplica a los elementos sintácticos de categoría "Todas" el valor numérico de categoría indicado en un cuadro de sintaxis en la posición de inclusión de una estructura sintáctica que contiene un elemento sintáctico con la categoría marcada como "Todas".

A continuación se describen los descriptores para el proceso de análisis sintáctico de cada elemento sintáctico. En algunos elementos sintácticos se utilizan dos descriptores separados por una barra vertical. En esos casos, se aplican los descriptores de la izquierda cuando `entropy_coding_mode_flag` es igual a 0 y los de la derecha cuando `entropy_coding_mode_flag` es igual a 1.

- `ae(v)`: elemento sintáctico codificado mediante codificación entrópica aritmética y adaptativa por contexto. El proceso de análisis sintáctico para este descriptor se describe en la subcláusula 9.3.
- `b(8)`: byte cuyo patrón es una cadena de bits (8 bits). El proceso de análisis sintáctico para este descriptor queda especificado por el valor que devuelve la función `read_bits( 8 )`.
- `ce(v)`: elemento sintáctico codificado mediante codificación entrópica de longitud variable y adaptativa por contexto, siendo el primer bit el izquierdo. El proceso de análisis sintáctico para este descriptor se describe en la subcláusula 9.2.
- `f(n)`: cadena de bits de patrón fijo que utiliza `n` bits escritos (de izquierda a derecha), siendo el primer bit el izquierdo. El proceso de análisis sintáctico para este descriptor queda especificado por el valor que devuelve la función `read_bits( n )`.
- `i(n)`: entero con signo de `n` bits. Cuando `n` es igual a "v" en el cuadro de sintaxis, el número de bits varía en función del valor de otros elementos sintácticos. El proceso de análisis sintáctico de este descriptor queda especificado por el valor que devuelve la función `read_bits( n )`, valor que se interpreta como la representación del número entero en complemento dos y de manera que el primer bit escrito es el más significativo.
- `me(v)`: elemento sintáctico codificado mediante codificación Exp-Golomb mapeado, siendo el primer bit el izquierdo. El proceso de análisis sintáctico para este descriptor se describe en la subcláusula 9.1.
- `se(v)`: elemento sintáctico codificado mediante codificación Exp-Golomb entero y con signo, siendo el primer bit el izquierdo. El proceso de análisis sintáctico para este descriptor se especifica en la subcláusula 9.1.
- `te(v)`: elemento sintáctico codificado mediante codificación Exp-Golomb truncada, siendo el primer bit el izquierdo. El proceso de análisis sintáctico para este descriptor se describe en la subcláusula 9.1.
- `u(n)`: entero sin signo de `n` bits. Cuando `n` es "v" en el cuadro de sintaxis, el número de bits varía en función del valor de otros elementos sintácticos. El proceso de análisis sintáctico para este descriptor queda especificado por el valor que devuelve la función `read_bits( n )`, valor que se interpreta como la representación binaria de un entero sin signo, siendo el bit más significativo el primero.
- `ue(v)`: elemento sintáctico codificado mediante codificación Exp-Golomb entero sin signo, siendo el primer bit el izquierdo. El proceso de análisis sintáctico para este descriptor se describe en la subcláusula 9.1.

### 7.3 Sintaxis en forma tabular

#### 7.3.1 Sintaxis de las unidades NAL

nal_unit( NumBytesInNALunit ) {	C	Descriptor
<b>forbidden_zero_bit</b>	Todas	f(1)
<b>nal_ref_idc</b>	Todas	u(2)
<b>nal_unit_type</b>	Todas	u(5)
NumBytesInRBSP = 0		
for( i = 1; i < NumBytesInNALunit; i++ ) {		
if( i + 2 < NumBytesInNALunit && next_bits( 24 ) == 0x000003 ) {		
<b>rbsp_byte</b> [ NumBytesInRBSP++ ]	Todas	b(8)
<b>rbsp_byte</b> [ NumBytesInRBSP++ ]	Todas	b(8)
i += 2		
<b>emulation_prevention_three_byte</b> /* igual a 0x03 */	Todas	f(8)
} else		
<b>rbsp_byte</b> [ NumBytesInRBSP++ ]	Todas	b(8)
}		
}		

### 7.3.2 Sintaxis de las cabidas útiles de secuencia en bytes en bruto y de sus bits de cola

#### 7.3.2.1 Sintaxis de la RBSP conjunto de parámetros secuencia

	C	Descriptor
seq_parameter_set_rbsp() {		
profile_idc	0	u(8)
constraint_set0_flag	0	u(1)
constraint_set1_flag	0	u(1)
constraint_set2_flag	0	u(1)
reserved_zero_5bits /* equal to 0 */	0	u(5)
level_idc	0	u(8)
seq_parameter_set_id	0	ue(v)
log2_max_frame_num_minus4	0	ue(v)
pic_order_cnt_type	0	ue(v)
if( pic_order_cnt_type == 0 )		
log2_max_pic_order_cnt_lsb_minus4	0	ue(v)
else if( pic_order_cnt_type == 1 ) {		
delta_pic_order_always_zero_flag	0	u(1)
offset_for_non_ref_pic	0	se(v)
offset_for_top_to_bottom_field	0	se(v)
num_ref_frames_in_pic_order_cnt_cycle	0	ue(v)
for( i=0; i < num_ref_frames_in_pic_order_cnt_cycle; i++ )		
offset_for_ref_frame[ i ]	0	se(v)
}		
num_ref_frames	0	ue(v)
gaps_in_frame_num_value_allowed_flag	0	u(1)
pic_width_in_mbs_minus1	0	ue(v)
pic_height_in_map_units_minus1	0	ue(v)
frame_mbs_only_flag	0	u(1)
if( !frame_mbs_only_flag )		
mb_adaptive_frame_field_flag	0	u(1)
direct_8x8_inference_flag	0	u(1)
frame_cropping_flag	0	u(1)
if( frame_cropping_flag ) {		
frame_crop_left_offset	0	ue(v)
frame_crop_right_offset	0	ue(v)
frame_crop_top_offset	0	ue(v)
frame_crop_bottom_offset	0	ue(v)
}		
vui_parameters_present_flag	0	u(1)
if( vui_parameters_present_flag )		
vui_parameters( )	0	
rbsp_trailing_bits( )	0	
}		

### 7.3.2.2 Sintaxis de la RBSP conjunto de parámetros imagen

	<b>C</b>	<b>Descriptor</b>
pic parameter set rbsp() {		
<b>pic parameter set id</b>	1	ue(v)
<b>seq parameter set id</b>	1	ue(v)
<b>entropy coding mode flag</b>	1	u(1)
<b>pic order present flag</b>	1	u(1)
<b>num slice groups minus1</b>	1	ue(v)
if( num slice groups minus1 > 0 ) {		
<b>slice group map type</b>	1	ue(v)
if( slice_group_map_type == 0 )		
for( iGroup = 0; iGroup <= num slice groups minus1; iGroup++ )		
<b>run_length_minus1[ iGroup ]</b>	1	ue(v)
else if( slice_group_map_type == 2 )		
for( iGroup = 0; iGroup < num slice groups minus1; iGroup++ ) {		
<b>top_left[ iGroup ]</b>	1	ue(v)
<b>bottom_right[ iGroup ]</b>	1	ue(v)
}		
else if( slice_group_map_type == 3    slice_group_map_type == 4    slice_group_map_type == 5 ) {		
<b>slice_group_change_direction_flag</b>	1	u(1)
<b>slice_group_change_rate_minus1</b>	1	ue(v)
} else if( slice_group_map_type == 6 ) {		
<b>pic_size_in_map_units_minus1</b>	1	ue(v)
for( i = 0; i <= pic_size_in_map_units_minus1; i++ )		
<b>slice_group_id[ i ]</b>	1	u(v)
}		
}		
}		
<b>num_ref_idx_l0_active_minus1</b>	1	ue(v)
<b>num_ref_idx_l1_active_minus1</b>	1	ue(v)
<b>weighted_pred_flag</b>	1	u(1)
<b>weighted_bipred_idc</b>	1	u(2)
<b>pic_init_qp_minus26</b> /* relativo a 26 */	1	se(v)
<b>pic_init_qs_minus26</b> /* relativo a 26 */	1	se(v)
<b>chroma_qp_index_offset</b>	1	se(v)
<b>deblocking_filter_control_present_flag</b>	1	u(1)
<b>constrained_intra_pred_flag</b>	1	u(1)
<b>redundant_pic_cnt_present_flag</b>	1	u(1)
rbsp_trailing_bits()	1	
}		

### 7.3.2.3 Sintaxis de la RBSP información de perfeccionamiento complementaria

	<b>C</b>	<b>Descriptor</b>
sei_rbsp() {		
do		
sei_message()	5	
while( more_rbsp_data() )		
rbsp_trailing_bits()	5	
}		

### 7.3.2.3.1 Sintaxis del mensaje de información de perfeccionamiento complementaria

	C	Descriptor
sei message() {		
payloadType = 0		
while( next_bits( 8 ) == 0xFF ) {		
<b>ff_byte</b> /* igual a 0xFF */	5	f(8)
payloadType += 255		
}		
<b>last_payload_type_byte</b>	5	u(8)
payloadType += last_payload_type_byte		
payloadSize = 0		
while( next_bits( 8 ) == 0xFF ) {		
<b>ff_byte</b> /* igual a 0xFF */	5	f(8)
payloadSize += 255		
}		
<b>last_payload_size_byte</b>	5	u(8)
payloadSize += last_payload_size_byte		
sei_payload( payloadType, payloadSize )	5	
}		

### 7.3.2.4 Sintaxis de la RBSP delimitador de unidades de acceso

	C	Descriptor
access_unit_delimiter_rbsp() {		
<b>primary_pic_type</b>	6	u(3)
rbsp_trailing_bits()	6	
}		

### 7.3.2.5 Sintaxis de la RBSP fin de secuencia

	C	Descriptor
end_of_seq_rbsp() {		
}		

### 7.3.2.6 Sintaxis de la RBSP fin de tren

	C	Descriptor
end_of_stream_rbsp() {		
}		

### 7.3.2.7 Sintaxis de la RBSP de datos de relleno

	C	Descriptor
filler_data_rbsp( NumBytesInRBSP ) {		
while( next_bits( 8 ) == 0xFF )		
<b>ff_byte</b> /* igual a 0xFF */	9	f(8)
rbsp_trailing_bits()	9	
}		



### 7.3.2.8 Sintaxis de la RBSP capa de sector sin particiones

	<b>C</b>	<b>Descriptor</b>
slice_layer_without_partitioning_rbsp() {		
slice_header()	2	
slice_data() /* todas las categorías de la sintaxis de slice_data() */	2   3   4	
rbsp_slice_trailing_bits()	2	
}		

### 7.3.2.9 Sintaxis de la RBSP particiones de datos de sector

#### 7.3.2.9.1 Sintaxis de la RBSP partición A de datos de sector

	<b>C</b>	<b>Descriptor</b>
slice_data_partition_a_layer_rbsp() {		
slice_header()	2	
<b>slice_id</b>	2	ue(v)
slice_data() /* sólo las partes de la categoría 2 de la sintaxis de slice_data() */	2	
rbsp_slice_trailing_bits()	2	
}		

#### 7.3.2.9.2 Sintaxis de la RBSP partición B de datos de sector

	<b>C</b>	<b>Descriptor</b>
slice_data_partition_b_layer_rbsp() {		
<b>slice_id</b>	3	ue(v)
if( redundant_pic_cnt_present_flag )		
<b>redundant_pic_cnt</b>	3	ue(v)
slice_data() /* sólo las partes de la categoría 3 de la sintaxis de slice_data() */	3	
rbsp_slice_trailing_bits()	3	
}		

#### 7.3.2.9.3 Sintaxis de la RBSP partición C de datos de sector

	<b>C</b>	<b>Descriptor</b>
slice_data_partition_c_layer_rbsp() {		
<b>slice_id</b>	4	ue(v)
if( redundant_pic_cnt_present_flag )		
<b>redundant_pic_cnt</b>	4	ue(v)
slice_data() /* sólo las partes de la categoría 4 de la sintaxis de slice_data() */	4	
rbsp_slice_trailing_bits()	4	
}		

### 7.3.2.10 Sintaxis de los bits de cola de sector de la RBSP

	<b>C</b>	<b>Descriptor</b>
rbsp_slice_trailing_bits() {		
rbsp_trailing_bits()	Todas	
if( entropy_coding_mode_flag )		
while( more_rbsp_trailing_data() )		
<b>cabac_zero_word</b> /* igual a 0x0000 */	Todas	f(16)
}		

### 7.3.2.11 Sintaxis de los bits de cola de la RBSP

	<b>C</b>	<b>Descriptor</b>
rbsp_trailing_bits() {		
<b>rbsp_stop_one_bit</b> /* igual a 1 */	Todas	f(1)
while( !byte_aligned() )		
<b>rbsp_alignment_zero_bit</b> /* igual a 0 */	Todas	f(1)
}		

### 7.3.3 Sintaxis de la cabecera de sector

	<b>C</b>	<b>Descriptor</b>
slice header() {		
<b>first mb in slice</b>	2	ue(v)
<b>slice type</b>	2	ue(v)
<b>pic parameter set id</b>	2	ue(v)
<b>frame num</b>	2	u(v)
if( !frame mbs only flag ) {		
<b>field pic flag</b>	2	u(1)
if( field pic flag )		
<b>bottom field flag</b>	2	u(1)
}		
if( nal unit type == 5 )		
<b>idr pic id</b>	2	ue(v)
if( pic order cnt type == 0 ) {		
<b>pic order cnt lsb</b>	2	u(v)
if( pic order present flag && !field pic flag )		
<b>delta pic order cnt bottom</b>	2	se(v)
}		
if( pic order cnt type == 1 && !delta pic order always zero flag ) {		
<b>delta pic order cnt[ 0 ]</b>	2	se(v)
if( pic order present flag && !field pic flag )		
<b>delta pic order cnt[ 1 ]</b>	2	se(v)
}		
if( redundant pic cnt present flag )		
<b>redundant pic cnt</b>	2	ue(v)
if( slice type == B )		
<b>direct spatial mv pred flag</b>	2	u(1)
if( slice type == P    slice type == SP    slice type == B ) {		
<b>num ref idx active override flag</b>	2	u(1)
if( num ref idx active override flag ) {		
<b>num ref idx l0 active minus1</b>	2	ue(v)
if( slice type == B )		
<b>num ref idx l1 active minus1</b>	2	ue(v)
}		
}		
ref pic list reordering()	2	
if( ( weighted_pred_flag && ( slice_type == P    slice_type == SP ) )    ( weighted_bipred_idc == 1 && slice_type == B ) )		
pred_weight_table()	2	
if( nal_ref_idc != 0 )		
dec_ref_pic_marking()	2	
if( entropy_coding_mode_flag && slice_type != I && slice_type != SI )		
<b>cabac_init_idc</b>	2	ue(v)
<b>slice_qp_delta</b>	2	se(v)
if( slice_type == SP    slice_type == SI ) {		
if( slice_type == SP )		
<b>sp_for_switch_flag</b>	2	u(1)
<b>slice_qs_delta</b>	2	se(v)
}		
if( deblocking_filter_control_present_flag ) {		
<b>disable_deblocking_filter_idc</b>	2	ue(v)
if( disable_deblocking_filter_idc != 1 ) {		
<b>slice_alpha_c0_offset_div2</b>	2	se(v)
<b>slice_beta_offset_div2</b>	2	se(v)
}		
if( num_slice_groups_minus1 > 0 && slice_group_map_type >= 3 && slice_group_map_type <= 5 )		
<b>slice_group_change_cycle</b>	2	u(v)
}		

### 7.3.3.1 Sintaxis de la reordenación de lista de imágenes de referencia

	<b>C</b>	<b>Descriptor</b>
ref pic list reordering() {		
if( slice type != I && slice type != SI ) {		
<b>ref pic list reordering flag I0</b>	2	u(1)
if( ref pic list reordering flag I0 )		
do {		
<b>reordering of pic nums idc</b>	2	ue(v)
if( reordering_of_pic_nums_idc == 0    reordering_of_pic_nums_idc == 1 )		
<b>abs diff pic num minus1</b>	2	ue(v)
else if( reordering_of_pic_nums_idc == 2 )		
<b>long term pic num</b>	2	ue(v)
} while( reordering_of_pic_nums_idc != 3 )		
}		
if( slice_type == B ) {		
<b>ref pic list reordering flag I1</b>	2	u(1)
if( ref pic list reordering flag I1 )		
do {		
<b>reordering of pic nums idc</b>	2	ue(v)
if( reordering_of_pic_nums_idc == 0    reordering_of_pic_nums_idc == 1 )		
<b>abs diff pic num minus1</b>	2	ue(v)
else if( reordering_of_pic_nums_idc == 2 )		
<b>long term pic num</b>	2	ue(v)
} while( reordering_of_pic_nums_idc != 3 )		
}		
}		
}		

### 7.3.3.2 Sintaxis del cuadro de ponderaciones de predicción

	<b>C</b>	<b>Descriptor</b>
pred_weight_table() {		
<b>luma_log2_weight_denom</b>	2	ue(v)
<b>chroma_log2_weight_denom</b>	2	ue(v)
for( i = 0; i <= num_ref_idx_l0_active_minus1; i++ ) {		
<b>luma_weight_l0_flag</b>	2	u(1)
if( luma_weight_l0_flag ) {		
<b>luma_weight_l0[ i ]</b>	2	se(v)
<b>luma_offset_l0[ i ]</b>	2	se(v)
}		
<b>chroma_weight_l0_flag</b>	2	u(1)
if( chroma_weight_l0_flag )		
for( j = 0; j < 2; j++ ) {		
<b>Chroma_weight_l0[ i ][ j ]</b>	2	se(v)
<b>Chroma_offset_l0[ i ][ j ]</b>	2	se(v)
}		
}		
if( slice_type == B )		
for( i = 0; i <= num_ref_idx_l1_active_minus1; i++ ) {		
<b>luma_weight_l1_flag</b>	2	u(1)
if( luma_weight_l1_flag ) {		
<b>luma_weight_l1[ i ]</b>	2	se(v)
<b>luma_offset_l1[ i ]</b>	2	se(v)
}		
<b>chroma_weight_l1_flag</b>	2	u(1)
if( chroma_weight_l1_flag )		
for( j = 0; j < 2; j++ ) {		
<b>chroma_weight_l1[ i ][ j ]</b>	2	se(v)
<b>chroma_offset_l1[ i ][ j ]</b>	2	se(v)
}		
}		
}		
}		

### 7.3.3.3 Sintaxis del marcado de imágenes de referencia decodificadas

	<b>C</b>	<b>Descriptor</b>
dec_ref_pic_marking() {		
if( nal_unit_type == 5 ) {		
<b>no_output_of_prior_pics_flag</b>	2   5	u(1)
<b>long_term_reference_flag</b>	2   5	u(1)
} else {		
<b>adaptive_ref_pic_marking_mode_flag</b>	2   5	u(1)
if( adaptive_ref_pic_marking_mode_flag )		
do {		
<b>memory_management_control_operation</b>	2   5	ue(v)
if( memory_management_control_operation == 1    memory_management_control_operation == 3 )		
<b>difference_of_pic_nums_minus1</b>	2   5	ue(v)
if( memory_management_control_operation == 2 )		
<b>Long_term_pic_num</b>	2   5	ue(v)
if( memory_management_control_operation == 3    memory_management_control_operation == 6 )		
<b>Long_term_frame_idx</b>	2   5	ue(v)
if( memory_management_control_operation == 4 )		
<b>Max_long_term_frame_idx_plus1</b>	2   5	ue(v)
} while( memory_management_control_operation != 0 )		
}		
}		

### 7.3.4 Sintaxis de los datos de sector

	C	Descriptor
slice data() {		
if( entropy_coding_mode_flag )		
while( !byte_aligned() )		
<b>cabac alignment one bit</b>	2	f(1)
CurrMbAddr = first_mb_in_slice * ( 1 + MbaffFrameFlag )		
moreDataFlag = 1		
prevMbSkipped = 0		
do {		
if( slice_type != I && slice_type != SI )		
if( !entropy_coding_mode_flag ) {		
<b>mb skip run</b>	2	ue(v)
prevMbSkipped = ( mb_skip_run > 0 )		
for( i=0; i<mb_skip_run; i++ )		
CurrMbAddr = NextMbAddress( CurrMbAddr )		
moreDataFlag = more_rbsp_data()		
} else {		
<b>mb skip flag</b>	2	ae(v)
moreDataFlag = !mb_skip_flag		
}		
if( moreDataFlag ) {		
if( MbaffFrameFlag && ( CurrMbAddr % 2 == 0    ( CurrMbAddr % 2 == 1 && prevMbSkipped ) ) )		
<b>mb field decoding flag</b>	2	u(1)   ae(v)
Macroblock_layer()	2   3   4	
}		
if( !entropy_coding_mode_flag )		
moreDataFlag = more_rbsp_data()		
else {		
if( slice_type != I && slice_type != SI )		
prevMbSkipped = mb_skip_flag		
if( MbaffFrameFlag && CurrMbAddr % 2 == 0 )		
moreDataFlag = 1		
else {		
<b>end of slice flag</b>	2	ae(v)
moreDataFlag = !end_of_slice_flag		
}		
}		
CurrMbAddr = NextMbAddress( CurrMbAddr )		
} while( moreDataFlag )		
}		

### 7.3.5 Sintaxis de la capa macrobloque

macroblock layer( ) {	C	Descriptor
<b>mb_type</b>	2	ue(v)   ae(v)
if( mb_type == I_PCM ) {		
while( !byte_aligned( )		
<b>pcm_alignment_zero_bit</b>	2	f(1)
for( i = 0; i < 256 * ChromaFormatFactor; i++)		
<b>pcm_byte[ i ]</b>	2	u(8)
} else {		
if( MbPartPredMode( mb_type, 0 ) != Intra_4x4 && MbPartPredMode( mb_type, 0 ) != Intra_16x16 && NumMbPart( mb_type ) == 4 )		
sub_mb_pred( mb_type )	2	
Else		
mb_pred( mb_type )	2	
if( MbPartPredMode( mb_type, 0 ) != Intra_16x16 )		
<b>coded_block_pattern</b>	2	me(v)   ae(v)
if( CodedBlockPatternLuma > 0    CodedBlockPatternChroma > 0    MbPartPredMode( mb_type, 0 ) == Intra_16x16 ) {		
<b>mb_qp_delta</b>	2	se(v)   ae(v)
residual( )	3   4	
}		
}		
}		
}		

#### 7.3.5.1 Sintaxis de la predicción de macrobloques

mb_pred( mb_type ) {	C	Descriptor
if( MbPartPredMode( mb_type, 0 ) == Intra_4x4    MbPartPredMode( mb_type, 0 ) == Intra_16x16 ) {		
if( MbPartPredMode( mb_type, 0 ) == Intra_4x4 )		
for( luma4x4BlkIdx=0; luma4x4BlkIdx<16; luma4x4BlkIdx++ ) {		
<b>prev_intra4x4_pred_mode_flag[ luma4x4BlkIdx ]</b>	2	u(1)   ae(v)
if( !prev_intra4x4_pred_mode_flag[ luma4x4BlkIdx ] )		
<b>rem_intra4x4_pred_mode[ luma4x4BlkIdx ]</b>	2	u(3)   ae(v)
}		
<b>intra_chroma_pred_mode</b>	2	ue(v)   ae(v)
} else if( MbPartPredMode( mb_type, 0 ) != Direct ) {		
for( mbPartIdx = 0; mbPartIdx < NumMbPart( mb_type ); mbPartIdx++)		
if( ( num_ref_idx_l0_active_minus1 > 0    mb_field_decoding_flag ) && MbPartPredMode( mb_type, mbPartIdx ) != Pred_L1 )		
<b>ref_idx_l0[ mbPartIdx ]</b>	2	te(v)   ae(v)
for( mbPartIdx = 0; mbPartIdx < NumMbPart( mb_type ); mbPartIdx++)		
if( ( num_ref_idx_l1_active_minus1 > 0    mb_field_decoding_flag ) && MbPartPredMode( mb_type, mbPartIdx ) != Pred_L0 )		
<b>ref_idx_l1[ mbPartIdx ]</b>	2	te(v)   ae(v)
for( mbPartIdx = 0; mbPartIdx < NumMbPart( mb_type ); mbPartIdx++)		
if( MbPartPredMode( mb_type, mbPartIdx ) != Pred_L1 )		
for( compIdx = 0; compIdx < 2; compIdx++)		
<b>mvd_l0[ mbPartIdx ][ 0 ][ compIdx ]</b>	2	se(v)   ae(v)
for( mbPartIdx = 0; mbPartIdx < NumMbPart( mb_type ); mbPartIdx++)		
if( MbPartPredMode( mb_type, mbPartIdx ) != Pred_L0 )		
for( compIdx = 0; compIdx < 2; compIdx++)		
<b>mvd_l1[ mbPartIdx ][ 0 ][ compIdx ]</b>	2	se(v)   ae(v)
}		
}		
}		
}		

### 7.3.5.2 Sintaxis de la predicción de submacrobloques

	C	Descriptor
sub mb_pred( mb_type ) {		
for( mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++ )		
<b>sub_mb_type</b> [ mbPartIdx ]	2	ue(v)   ae(v)
for( mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++ )		
if( ( num_ref_idx_l0_active_minus1 > 0    mb_field_decoding_flag ) && mb_type != P_8x8ref0 && sub_mb_type[ mbPartIdx ] != B_Direct_8x8 && SubMbPredMode( sub_mb_type[ mbPartIdx ] ) != Pred_L1 )		
<b>ref_idx_l0</b> [ mbPartIdx ]	2	te(v)   ae(v)
for( mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++ )		
if( ( num_ref_idx_l1_active_minus1 > 0    mb_field_decoding_flag ) && sub_mb_type[ mbPartIdx ] != B_Direct_8x8 && SubMbPredMode( sub_mb_type[ mbPartIdx ] ) != Pred_L0 )		
<b>ref_idx_l1</b> [ mbPartIdx ]	2	te(v)   ae(v)
for( mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++ )		
if( sub_mb_type[ mbPartIdx ] != B_Direct_8x8 && SubMbPredMode( sub_mb_type[ mbPartIdx ] ) != Pred_L1 )		
for( subMbPartIdx = 0; subMbPartIdx < NumSubMbPart( sub_mb_type[ mbPartIdx ] ); subMbPartIdx++ )		
for( compIdx = 0; compIdx < 2; compIdx++ )		
<b>mvd_l0</b> [ mbPartIdx ][ subMbPartIdx ][ compIdx ]	2	se(v)   ae(v)
for( mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++ )		
if( sub_mb_type[ mbPartIdx ] != B_Direct_8x8 && SubMbPredMode( sub_mb_type[ mbPartIdx ] ) != Pred_L0 )		
for( subMbPartIdx = 0; subMbPartIdx < NumSubMbPart( sub_mb_type[ mbPartIdx ] ); subMbPartIdx++ )		
for( compIdx = 0; compIdx < 2; compIdx++ )		
<b>mvd_l1</b> [ mbPartIdx ][ subMbPartIdx ][ compIdx ]	2	se(v)   ae(v)
}		



### 7.3.5.3 Sintaxis de los datos residuales

residual( ) {	C	Descriptor
if( !entropy coding mode flag )		
residual_block = residual_block_cavlc		
else		
residual_block = residual_block_cabac		
if( MbPartPredMode( mb_type, 0 ) == Intra_16x16 )		
residual_block( Intra16x16DCLevel, 16 )	3	
for( i8x8 = 0; i8x8 < 4; i8x8++ ) /* cada bloque 8x8 luma */		
for( i4x4 = 0; i4x4 < 4; i4x4++ ) /* cada subbloque 4x4 del bloque */		
if( CodedBlockPatternLuma & ( 1 << i8x8 ) ) {		
if( MbPartPredMode( mb_type, 0 ) == Intra_16x16 )		
residual_block( Intra16x16ACLevel[ i8x8 * 4 + i4x4 ], 15 )	3	
else		
residual_block( LumaLevel[ i8x8 * 4 + i4x4 ], 16 )	3   4	
} else {		
if( MbPartPredMode( mb_type, 0 ) == Intra_16x16 )		
for( i = 0; i < 15; i++ )		
Intra16x16ACLevel[ i8x8 * 4 + i4x4 ][ i ] = 0		
else		
for( i = 0; i < 16; i++ )		
LumaLevel[ i8x8 * 4 + i4x4 ][ i ] = 0		
}		
for( iCbCr = 0; iCbCr < 2; iCbCr++ )		
if( CodedBlockPatternChroma & 3 ) /* hay residuo del nivel continuo cromina */		
residual_block( ChromaDCLevel[ iCbCr ], 4 )	3   4	
else		
for( i = 0; i < 4; i++ )		
ChromaDCLevel[ iCbCr ][ i ] = 0		
for( iCbCr = 0; iCbCr < 2; iCbCr++ )		
for( i4x4 = 0; i4x4 < 4; i4x4++ )		
if( CodedBlockPatternChroma & 2 )		
/* hay residuo de nivel continuo cromina */		
residual_block( ChromaACLevel[ iCbCr ][ i4x4 ], 15 )	3   4	
else		
for( i = 0; i < 15; i++ )		
ChromaACLevel[ iCbCr ][ i4x4 ][ i ] = 0		
}		

### 7.3.5.3.1 Sintaxis de la CAVLC de bloques residuales

	C	Descriptor
residual_block_cavlc( coeffLevel, maxNumCoeff ) {		
for( i = 0; i < maxNumCoeff; i++ )		
coeffLevel[ i ] = 0		
<b>coeff_token</b>	3   4	ce(v)
if( TotalCoeff( coeff_token ) > 0 ) {		
if( TotalCoeff( coeff_token ) > 10 && TrailingOnes( coeff_token ) < 3 )		
suffixLength = 1		
else		
suffixLength = 0		
for( i = 0; i < TotalCoeff( coeff_token ); i++ )		
if( i < TrailingOnes( coeff_token ) ) {		
<b>trailing_ones_sign_flag</b>	3   4	u(1)
level[ i ] = 1 - 2 * trailing_ones_sign_flag		
} else {		
<b>level_prefix</b>	3   4	ce(v)
levelCode = ( level_prefix << suffixLength )		
if( suffixLength > 0    level_prefix >= 14 ) {		
<b>level_suffix</b>	3   4	u(v)
levelCode += level_suffix		
}		
if( level_prefix == 15 && suffixLength == 0 )		
levelCode += 15		
if( i == TrailingOnes( coeff_token ) &&		
TrailingOnes( coeff_token ) < 3 )		
levelCode += 2		
if( levelCode % 2 == 0 )		
level[ i ] = ( levelCode + 2 ) >> 1		
else		
level[ i ] = ( -levelCode - 1 ) >> 1		
if( suffixLength == 0 )		
suffixLength = 1		
if( Abs( level[ i ] ) > ( 3 << ( suffixLength - 1 ) ) &&		
suffixLength < 6 )		
suffixLength++		
}		
if( TotalCoeff( coeff_token ) < maxNumCoeff ) {		
<b>total_zeros</b>	3   4	ce(v)
zerosLeft = total_zeros		
} else		
zerosLeft = 0		
for( i = 0; i < TotalCoeff( coeff_token ) - 1; i++ ) {		
if( zerosLeft > 0 ) {		
<b>run_before</b>	3   4	ce(v)
run[ i ] = run_before		
} else		
run[ i ] = 0		
zerosLeft = zerosLeft - run[ i ]		
}		
run[ TotalCoeff( coeff_token ) - 1 ] = zerosLeft		
coeffNum = -1		
for( i = TotalCoeff( coeff_token ) - 1; i >= 0; i-- ) {		
coeffNum += run[ i ] + 1		
coeffLevel[ coeffNum ] = level[ i ]		
}		
}		
}		
}		

### 7.3.5.3.2 Sintaxis de la CABAC de bloques residuales

	C	Descriptor
<code>residual_block_cabac( coeffLevel, maxNumCoeff ) {</code>		
<b>coded_block_flag</b>	3   4	ae(v)
if( coded_block_flag ) {		
numCoeff = maxNumCoeff		
i = 0		
do {		
<b>significant_coeff_flag[ i ]</b>	3   4	ae(v)
if( significant_coeff_flag[ i ] ) {		
<b>last_significant_coeff_flag[ i ]</b>	3   4	ae(v)
if( last_significant_coeff_flag[ i ] ) {		
numCoeff = i + 1		
for( j = numCoeff; j < maxNumCoeff; j++ )		
coeffLevel[ j ] = 0		
}		
}		
i++		
} while( i < numCoeff-1 )		
<b>coeff_abs_level_minus1[ numCoeff-1 ]</b>	3   4	ae(v)
<b>coeff_sign_flag[ numCoeff-1 ]</b>	3   4	ae(v)
coeffLevel[ numCoeff-1 ] =		
( coeff_abs_level_minus1[ numCoeff - 1 ] + 1 ) *		
( 1 - 2 * coeff_sign_flag[ numCoeff - 1 ] )		
for( i = numCoeff-2; i >= 0; i-- ) {		
if( significant_coeff_flag[ i ] ) {		
<b>coeff_abs_level_minus1[ i ]</b>	3   4	ae(v)
<b>coeff_sign_flag[ i ]</b>	3   4	ae(v)
coeffLevel[ i ] = ( coeff_abs_level_minus1[ i ] + 1 ) *		
( 1 - 2 * coeff_sign_flag[ i ] )		
} else		
coeffLevel[ i ] = 0		
}		
} else		
for( i = 0; i < maxNumCoeff; i++ )		
coeffLevel[ i ] = 0		
}		

## 7.4 Semántica

### 7.4.1 Semántica de las unidades NAL

NOTA – La VCL representa eficientemente el contenido de los datos vídeo. La NAL da formato a esos datos y proporciona información de cabecera adaptada al transporte por diversos canales de comunicación o el almacenamiento en diferentes medios. Todos los datos se incluyen en unidades NAL, las cuales contienen un número entero de bytes. La unidad NAL tiene un formato genérico que se puede utilizar en sistemas por paquetes y en sistemas de tren de bits. La diferencia estriba en que en el formato tren de bytes cada unidad NAL puede tener un prefijo código de inicio y bytes de relleno adicionales.

NumBytesInNALunit indica el tamaño de la unidad NAL en bytes. Este valor es necesario para decodificar la unidad NAL. Es necesario delimitar de alguna manera las unidades NAL para poder deducir NumBytesInNALunit. En el anexo B se especifica un método de delimitación para el formato tren de bytes. Otros posibles métodos de delimitación quedan fuera del alcance de esta Recomendación | Norma Internacional.

**forbidden\_zero\_bit** será igual a 0.

**nal\_ref\_idc** distinto de 0 indica que el contenido de la unidad NAL contiene un conjunto de parámetros secuencia o un conjunto de parámetros imagen o un sector de una imagen de referencia o una partición de datos de sector de una imagen de referencia.

nal\_ref\_idc igual a 0 para una unidad NAL que contiene un sector o una partición de datos de sector, indica que el sector o la partición de datos de sector forma parte de una imagen que no es de referencia.

nal\_ref\_idc será distinto de 0 en las unidades NAL que contienen un conjunto de parámetros secuencia o un conjunto de parámetros imagen. Cuando nal\_ref\_idc es igual a 0 para una unidad NAL de un sector o de una partición de datos de sector de una determinada imagen, será igual a 0 para todas las unidades NAL del sector o partición de datos de sector de la imagen.

nal\_ref\_idc será distinto de 0 para las unidades NAL IDR, es decir, unidades NAL con nal\_unit\_type igual a 5.

nal\_ref\_idc será igual a 0 para todas las unidades NAL cuyo nal\_unit\_type sea igual a 6, 9, 10, 11, ó 12.

**nal\_unit\_type** especifica el tipo de estructura de datos de la RBSP contenida en la unidad NAL, según se especifica en el cuadro 7-1. Las unidades NAL VCL se definen como aquellas unidades NAL cuyo nal\_unit\_type está entre 1 y 5, inclusive. Las demás unidades NAL se denominan unidades NAL no VCL.

La columna "C" en el cuadro 7-1 enumera las categorías de elementos sintácticos que pueden aparecer en la unidad NAL. Además, puede haber elementos sintácticos cuya categoría sintáctica sea "todas", según la sintaxis y la semántica de la estructura de datos RBSP. La presencia o no de elementos sintácticos en una determinada categoría enumerada depende de la sintaxis y la semántica de la estructura de datos RBSP correspondiente. El nal\_unit\_type será distinto de 3 y 4 salvo cuando haya algún elemento sintáctico en la estructura de datos RBSP que tenga una categoría de elemento sintáctico de valor igual al de nal\_unit\_type y su categoría no sea "todas".

**Cuadro 7-1 – Códigos de tipos de unidades NAL**

<b>nal_unit_type</b>	<b>Contenido de la unidad NAL y de la estructura sintáctica RBSP</b>	<b>C</b>
0	No especificado	
1	Sector codificado de una imagen que no es IDR slice_layer_without_partitioning_rbsp( )	2, 3, 4
2	Partición A de datos de sector codificado slice_data_partition_a_layer_rbsp( )	2
3	Partición B de datos de sector codificado slice_data_partition_b_layer_rbsp( )	3
4	Partición C de datos de sector codificado slice_data_partition_c_layer_rbsp( )	4
5	Sector codificado de una imagen IDR slice_layer_without_partitioning_rbsp( )	2, 3
6	Información de mejora suplementaria (SEI) sei_rbsp( )	5
7	Conjunto de parámetros secuencia seq_parameter_set_rbsp( )	0
8	Conjunto de parámetros imagen pic_parameter_set_rbsp( )	1
9	Delimitador de unidades de acceso access_unit_delimiter_rbsp( )	6
10	Fin de secuencia end_of_seq_rbsp( )	7
11	Fin de tren end_of_stream_rbsp( )	8
12	Datos de relleno filler_data_rbsp( )	9
13..23	Reservado	
24..31	No especificado	

Las unidades NAL cuyo nal\_unit\_type sea igual a 0 o esté en la gama de 24..31, inclusive, no afectarán al proceso de decodificación especificado en esta Recomendación | Norma Internacional.

NOTA – La utilización de los tipos de unidades NAL 0 y 24..31 es específico de la aplicación. En esta Recomendación | Norma Internacional no se especifica el proceso de decodificación para estos valores de nal\_unit\_type.

Los decodificadores harán caso omiso (eliminarán del tren de bits y descartarán) del contenido de todas las unidades NAL que utilicen valores reservados de nal\_unit\_type.

NOTA – De este modo se podrá definir en el futuro extensiones compatibles con esta Recomendación | Norma Internacional.

En el texto, por unidad NAL de sector codificado se entiende un sector codificado de una unidad NAL de imagen que no es IDR o un sector codificado de una unidad NAL de imagen IDR.

Cuando el valor de nal\_unit\_type es 5 para una unidad NAL que contiene un sector de una imagen codificada, el valor de nal\_unit\_type será 5 en todas las demás unidades NAL VCL de la misma imagen codificada. Las imágenes de este tipo se denominan imágenes IDR.

NOTA – No se pueden utilizar las particiones de datos de sector en imágenes IDR.

**rbsp\_byte[ i ]** es el i-ésimo byte de una RBSP. Las RBSP se definen como una secuencia ordenada de bytes, del modo siguiente.

La RBSP contiene una SODB de modo que:

- Si la SODB está vacía (es decir, su longitud es de cero bits), la RBSP también está vacía.
- De lo contrario, la RBSP contiene la SODB de modo que:
  - 1) El primer byte de la RBSP contiene los ocho bits de la SODB (siendo el más significativo el de más a la izquierda); el siguiente byte de la RBSP contiene los siguientes ocho bits de la SODB, etc., hasta que queden menos de ocho bits de la SODB.
  - 2) **rbsp\_trailing\_bits( )** está presente después de la SODB del modo siguiente:
    - i) los primeros bits (siendo el más significativo el de más a la izquierda) del último byte de la RBSP contienen los bits restantes de la SODB (si los hubiere),
    - ii) el siguiente bit consta de un solo **rbsp\_stop\_one\_bit** igual a 1, y
    - iii) cuando el **rbsp\_stop\_one\_bit** no es el último bit del byte alineado por byte, habrá algún **rbsp\_alignment\_zero\_bit** para realizar el alineamiento de byte.
  - 3) Es posible que haya algún elemento sintáctico de 16 bits **cabac\_zero\_word** igual a 0x0000 en algunas RBSP después de **rbsp\_trailing\_bits( )** al final de la RBSP.

Las estructuras sintácticas que tienen estas propiedades RBSP se indican mediante el sufijo "**\_rbsp**" en los cuadros de sintaxis. Estas estructuras estarán contenidas en los bytes de datos **rbsp\_byte[ i ]** de las unidades NAL. La asociación de estructuras sintácticas RBSP con las unidades NAL se especifica en el cuadro 7-1.

NOTA – Cuando se conocen los límites de la RBSP, el decodificador puede extraer la SODB de la RBSP concatenando los bits de los bytes de la RBSP y descartando el **rbsp\_stop\_one\_bit**, que es el último bit (el menos significativo, el que está más a la derecha) igual a 1, y descartará los siguientes bits (los menos significativos, más a la derecha), que serán todos 0. Los datos necesarios para el proceso de decodificación están contenidos en la parte SODB de la RBSP.

**emulation\_prevention\_three\_byte** es un byte de valor 0x03. Si hay un **emulation\_prevention\_three\_byte** en la unidad NAL, el proceso de decodificación lo descartará.

El último byte de la unidad NAL no será igual a 0x00.

En la unidad NAL no aparecerá ninguna de las siguientes secuencias de tres bytes en posición alguna alineada por byte:

- 0x000000
- 0x000001
- 0x000002

En la unidad NAL cualquier secuencia de cuatro bytes que comience con 0x000003 distintas de las siguientes no podrán aparecer en posición alguna alineada por byte:

- 0x00000300
- 0x00000301
- 0x00000302
- 0x00000303

#### **7.4.1.1 Encapsulado de una SODB dentro de una RBSP (informativa)**

Esta subcláusula no es parte integrante de esta Recomendación | Norma Internacional.

A continuación se especifica el encapsulado de una SODB dentro de una RBSP y la utilización del `emulation_prevention_three_byte` para el encapsulado de una RBSP dentro de una unidad NAL, con objeto de:

- evitar la emulación de códigos de inicio dentro de unidades NAL y además permitir la representación de cualquier SODB dentro de una unidad NAL,
- permitir la identificación del final de la SODB dentro de la unidad NAL mediante la búsqueda en la RBSP del `rbsp_stop_one_bit` que aparece al final de la RBSP, y
- permitir que la unidad NAL tenga un tamaño mayor que la SODB en ciertas circunstancias (para lo cual se utilizará alguna `cabac_zero_word`).

El codificador puede producir la unidad NAL a partir de la RBSP mediante el siguiente procedimiento:

Se busca en los datos de la RBSP los bits alineados por byte que tengan el siguiente patrón binario:

'00000000 00000000 000000xx' (donde xx representa cualquier patrón de 2 bits: 00, 01, 10, u 11),

y se inserta el byte 0x03 de modo que el patrón de bits queda

'00000000 00000000 00000011 000000xx',

y, por último, cuando el último byte de los datos de la RBSP sea igual a 0x00 (lo cual sólo puede ocurrir cuando la RBSP termina en una `cabac_zero_word`) se añade al final de los datos el byte 0x03.

Se inserta al principio de la secuencia de bytes resultantes el primer byte de la unidad NAL que contiene la indicación del tipo de estructura de datos RBSP contenida. De este modo se da formato a toda la unidad NAL.

Mediante este proceso es posible representar cualquier SODB en una unidad NAL garantizando además que

- no se produce emulación de un prefijo de código de inicio alineado por byte dentro de la unidad NAL, y
- no se produce emulación de secuencia alguna de 8 bits de valor cero seguidos de un prefijo de código de inicio, independientemente de si están o no alineados por byte, dentro de la unidad NAL.

#### **7.4.1.2 Orden de unidades NAL y asociación con imágenes codificadas, unidades de acceso, y secuencias de vídeo**

En esta subcláusula se especifican las restricciones del orden de unidades NAL en el tren de bits. Los órdenes de unidades NAL en el tren de bits que obedezcan a estas pautas se denominarán en el documento como orden de decodificación de unidades NAL. Dentro de una unidad NAL, la sintaxis de las subcláusulas 7.3, D.1 y E.1 especifican el orden de decodificación de elementos sintácticos. Los decodificadores conformes con esta Recomendación | Norma Internacional serán capaces de recibir unidades NAL y sus elementos sintácticos en el orden de decodificación.

##### **7.4.1.2.1 Orden de las RBSP de conjunto de parámetros imagen y secuencia y su activación**

NOTA – El mecanismo de conjunto de parámetros imagen y secuencia separa la transmisión de información que cambia con poca frecuencia de la transmisión de datos de macrobloques codificados. En algunas aplicaciones, los conjuntos de parámetros imagen y secuencia se pueden transmitir "fuera de banda" utilizando un mecanismo de transporte fiable.

La RBSP del conjunto de parámetros imagen incluye parámetros a los cuales se puede hacer referencia en unidades NAL del sector codificado o en unidades NAL de la partición A de datos del sector codificado de una o más imágenes codificadas. Al comienzo del proceso de decodificación se

considera no activa cada RBSP del conjunto de parámetros imagen. En cualquier momento dado durante el proceso de decodificación se considera activa a lo sumo una RBSP del conjunto de parámetros imagen, y la activación de cualquier RBSP del conjunto de parámetros imagen particular da como resultado la desactivación de la RBSP del conjunto de parámetros imagen que se encontraba previamente activa (si la había).

Cuando se hace referencia a una RBSP, que no está activa, de conjunto de parámetros imagen (con un determinado valor de `pic_parameter_set_id`) mediante una unidad NAL de sector codificado o una unidad NAL de partición de datos de sector codificado (mediante ese valor de `pic_parameter_set_id`), se activa esta RBSP. Esta RBSP de conjunto de parámetros imagen es la RBSP de conjunto de parámetros imagen activa hasta que queda desactivada por la activación de otra RBSP de conjunto de parámetros imagen. Las RBSP de conjunto de parámetros imagen, con ese determinado valor de `pic_parameter_set_id`, deberán estar disponibles para el proceso de decodificación antes de su activación.

Toda unidad NAL de conjunto de parámetros imagen que contenga el valor de `pic_parameter_set_id` correspondiente a la RBSP de conjuntos de parámetros imagen activa deberá tener el mismo contenido que la de la RBSP de conjunto de parámetros imagen activa, a no ser que esté después de la última unidad NAL VCL y antes de la primera unidad NAL VCL de otra imagen codificada.

La RBSP de conjunto de parámetros secuencia incluye parámetros a los cuales se puede hacer referencia en una o más RBSP de conjunto de parámetros imagen o en una o más unidades NAL SEI que contienen un mensaje SEI de periodo de almacenamiento intermedio. Al comienzo del proceso de decodificación se considera no activa cada RBSP del conjunto de parámetros de secuencia. En cualquier momento dado durante el proceso de decodificación se considera activa a lo sumo una RBSP del conjunto de parámetros de secuencia, y la activación de cualquier RBSP del conjunto de parámetros de secuencia da como resultado la desactivación de la RBSP del conjunto de parámetros de secuencia que estaba previamente activa (si la había).

Cuando se hace referencia a una RBSP de conjunto de parámetros secuencia (con un determinado valor de `seq_parameter_set_id`) que no está activa mediante la activación de una RBSP de conjunto de parámetros imagen (que utiliza ese valor de `seq_parameter_set_id`) o en una unidad NAL SEI que contiene un mensaje SEI de periodo de almacenamiento intermedio (que utiliza ese valor de `seq_parameter_set_id`), se activa esa RBSP. Esta RBSP de conjunto de parámetros secuencia se conoce como la RBSP de conjunto de parámetros secuencia activa hasta que quede desactivada por la activación de otra RBSP de conjunto de parámetros secuencia. Las RBSP de conjuntos de parámetros de secuencia, con un determinado valor de `seq_parameter_set_id`, estarán disponibles para el proceso de decodificación antes de su activación. La RBSP de conjunto de parámetros de secuencia activa permanecerá activa para toda la secuencia de vídeo codificado.

NOTA – Debido a que una unidad de acceso IDR inicia una nueva secuencia de vídeo codificado y a que una RBSP del conjunto de parámetros de secuencia activa debe permanecer activa durante toda la secuencia de vídeo codificado, un mensaje SEI del periodo de almacenamiento puede activar una RBSP del conjunto de parámetros de secuencia únicamente cuando el mensaje SEI del periodo de almacenamiento forma parte de una unidad de acceso IDR.

Toda unidad NAL de conjunto de parámetros secuencia que contenga el valor de `seq_parameter_set_id` correspondiente a la RBSP de conjunto de parámetros secuencia activa deberá tener el mismo contenido que la RBSP de conjunto de parámetros secuencia activa, a no ser que esté después de la última unidad de acceso de una secuencia vídeo codificada y antes de la primera unidad NAL VCL y de la primera unidad NAL SEI que contiene un mensaje SEI de periodo de almacenamiento intermedio (si lo hubiera) de otra secuencia de vídeo codificado.

NOTA – Si la RBSP de conjunto de parámetros imagen o la RBSP de conjunto de parámetros secuencia se transmite dentro del tren de bits, estas restricciones imponen unas pautas aplicables al orden de unidades NAL que contienen la RBSP de conjunto de parámetros imagen o la RBSP de



conjunto de parámetros secuencia, respectivamente. De lo contrario (es decir la RBSP de conjunto de parámetro imagen o la RBSP de conjunto de parámetros secuencia se transmiten por otros medios no especificados en esta Recomendación | Norma Internacional), deberán estar disponibles para el proceso de decodificación en el momento adecuado de manera que se sigan estas pautas.

Toda restricción que recaiga sobre la relación entre los valores de los elementos sintácticos (y los valores de las variables que se derivan de esos elementos sintácticos) de los conjuntos de parámetros de secuencia y de los conjuntos de parámetros de imagen, con otros elementos sintácticos, es una restricción que se aplica sólo a los conjuntos activos de parámetros de imagen y de parámetros de secuencia. Si está presente una RBSP del conjunto de parámetros de secuencia que no se activa en el tren de bits sus elementos sintácticos tendrán valores que deben cumplir con las restricciones especificadas, al igual que si ésta es activada por referencia en otro tren de bits de conformación. Si está presente una RBSP del conjunto de parámetros de imagen que no se activa en el tren de bits sus elementos sintácticos tendrán valores que deben cumplir con las restricciones especificadas, al igual que si ésta es activada por referencia en otro tren de bits de conformación.

Durante el funcionamiento del proceso de decodificación (véase cláusula 8), los valores de los parámetros del conjunto de parámetros imagen activo y del conjunto de parámetros secuencia activo se considerarán válidos. Para interpretar los mensajes SEI, los valores de los parámetros del conjunto de parámetros imagen y del conjunto de parámetros secuencia que están activos para aplicar el proceso de decodificación de unidades NAL VCL de la imagen codificada primaria en la misma unidad de acceso se considerarán válidos a no ser que se especifique otra cosa en la semántica del mensaje SEI.

#### **7.4.1.2.2 Orden de unidades de acceso y asociación a secuencias de vídeo codificado**

Los trenes de bits conformes con esta Recomendación | Norma Internacional constan de una o más secuencias de vídeo codificado.

Una secuencia de vídeo codificado está formada de una o más unidades de acceso. El orden de las unidades NAL y de las imágenes codificadas y su relación con las unidades de acceso se describe en la subcláusula 7.4.1.2.3.

La primera unidad de acceso de cada secuencia de vídeo codificado es una unidad de acceso IDR. Todas las subsiguientes unidades de acceso en la secuencia de vídeo codificado son unidades de acceso que no son IDR.

El número de orden de imágenes correspondiente a las imágenes codificadas en unidades de acceso consecutivas en el orden de decodificación que contienen imágenes que no son de referencia nunca será decreciente.

La unidad de acceso, si la hubiere, que esté después de la unidad de acceso que contiene una unidad NAL fin de secuencia será una unidad de acceso IDR.

Cuando una unidad NAL SEI contiene datos que pertenecen a más de una unidad de acceso (por ejemplo, cuando el alcance de la unidad NAL SEI abarca a toda una secuencia de vídeo codificado) estará contenida en la primera unidad de acceso a la cual se aplica.

Cuando haya una unidad NAL fin de tren en una unidad de acceso, esa unidad de acceso será la última unidad de acceso en el tren de bits y la unidad NAL fin de tren estará en la última unidad NAL de esa unidad de acceso.

#### **7.4.1.2.3 Orden de unidades NAL y de imágenes codificadas y asociación con unidades de acceso**

Una unidad de acceso está formada por una imagen codificada primaria, con quizá algunas de sus correspondientes imágenes codificadas redundantes y tal vez con algunas unidades NAL que no son VCL. La asociación de unidades NAL VCL a imágenes codificadas primarias o redundantes se describe en la subcláusula 7.4.1.2.5.

Si la primera unidad NAL después de la última unidad NAL VCL de una imagen codificada primaria es una de las siguientes, ésta indica el principio de una nueva unidad de acceso.

- unidad NAL delimitadora de unidad de acceso (si la hubiere);
- unidad NAL de conjunto de parámetros secuencia (si la hubiere);
- unidad NAL de conjunto de parámetros imagen (si la hubiere);
- unidad NAL SEI (si la hubiere);
- unidades NAL cuyo `nal_unit_type` está entre 13 y 18, inclusive;
- la primera unidad NAL VCL de una imagen codificada primaria (siempre estará presente).

Las pautas para detectar la primera unidad NAL VCL de una imagen codificada primaria se especifican en la subcláusula 7.4.1.2.4.

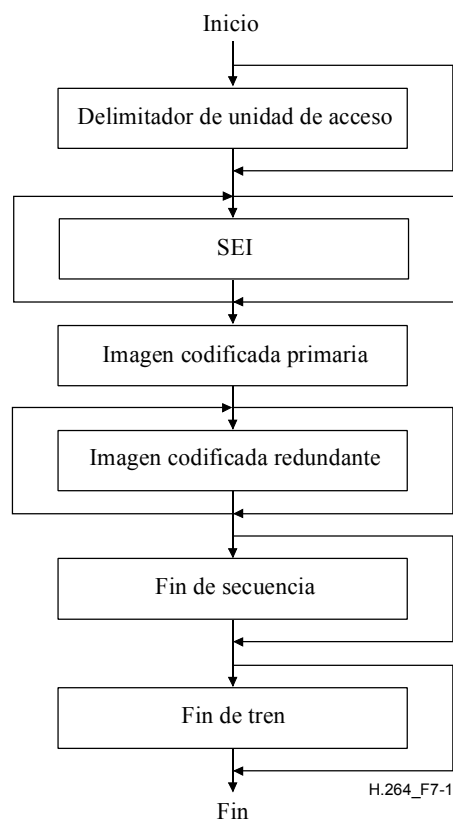
El orden de las imágenes codificadas y de unidades NAL distintas de las VCL dentro de una unidad de acceso deberá obedecer las siguientes pautas:

- Cuando haya una unidad NAL delimitadora de unidad de acceso, ésta será la primera unidad NAL. Toda unidad de acceso tendrá a lo sumo una unidad NAL delimitadora de unidad de acceso.
- Cuando haya unidades NAL SEI, estarán antes de la imagen codificada primaria.
- Cuando haya una unidad NAL SEI que contenga un mensaje SEI de periodo de almacenamiento intermedio, el mensaje SEI de periodo de almacenamiento intermedio será la primera cabida útil del mensaje SEI de la primera unidad NAL SEI en la unidad de acceso.
- La imagen codificada primaria estará antes que las correspondientes imágenes codificadas redundantes.
- Cuando haya imágenes codificadas redundantes estarán ordenadas en orden ascendente del valor de `redundant_pic_cnt`.
- Cuando haya una unidad NAL fin de secuencia, estará después de la imagen codificada primaria y de todas las imágenes codificadas redundantes (si las hubiere).
- Cuando haya una unidad NAL fin de tren, ésta deberá ser la última NAL.
- Las unidades NAL cuyo `nal_unit_type` sea igual a 0, 12, o esté entre 19 y 31, inclusive, no estarán antes que la primera unidad NAL VCL de la imagen codificada primaria.

NOTA – Si bien puede haber unidades NAL de conjunto de parámetros secuencia o unidades NAL de conjunto de parámetros imagen en una unidad de acceso, éstas no podrán estar después de la última unidad NAL VCL de la imagen codificada primaria dentro de la unidad de acceso, ya que esto indicaría el principio de una nueva unidad de acceso.

NOTA – Cuando haya una unidad NAL cuyo `nal_unit_type` sea igual a 7 u 8 en una unidad de acceso, es posible que no haga referencia a ella en las imágenes codificadas de la unidad de acceso en la que está ubicada, sino en las imágenes codificadas de subsiguientes unidades de accesos.

En la figura 7-1 se muestra la estructura de unidades de acceso que no contienen unidades NAL cuyo `nal_unit_type` es igual a 0, 7, 8, o está entre 12 y 31, inclusive.



**Figura 7-1 – Estructura de una unidad de acceso que no contiene unidades NAL cuyos nal\_unit\_type sean 0, 7, 8 o estén entre 12 y 31, inclusive**

#### 7.4.1.2.4 Detección de la primera unidad NAL VCL de una imagen codificada primaria

Esta subcláusula especifica los condicionantes a la sintaxis de la unidad NAL VCL, suficientes para permitir la detección de la primera unidad NAL VCL de cada imagen codificada primaria.

Toda unidad NAL de sector codificado o unidad NAL de la partición A de datos de sector codificado de una imagen codificada primaria de la unidad de acceso considerada se diferencia de cualquier unidad NAL de sector codificado o de cualquier unidad NAL de la partición A de datos de sector codificado de la imagen codificada primaria de la unidad de acceso previa en al menos una de las maneras siguientes.

- frame\_num tiene un valor diferente. El valor de frame\_num utilizado para verificar esta condición es el valor de frame\_num que aparece en la sintaxis de la cabecera de sector, sin importar si se infiere que ese valor ha sido igual a 0 para uso subsiguiente en el proceso de decodificación debido a la presencia de memory\_management\_control\_operation igual a 5.  
NOTA – Una consecuencia de la afirmación anterior es que una imagen codificada primaria, con frame\_num igual a 1, no puede tener una memory\_management\_control\_operation igual a 5, a no ser que en la imagen codificada primaria que le sigue (si la hay), se cumpla alguna de las otras condiciones enumeradas a continuación.
- pic\_parameter\_set\_id tiene un valor diferente.
- field\_pic\_flag tiene un valor diferente.
- bottom\_field\_flag está en ambas y tiene valores distintos.
- nal\_ref\_idc tiene un valor diferente, y uno de los valores de nal\_ref\_idc es igual a 0.
- pic\_order\_cnt\_type es igual a 0 en ambas y, o bien pic\_order\_cnt\_lsb tiene un valor diferente, o delta\_pic\_order\_cnt\_bottom tiene un valor diferente.

- pic\_order\_cnt\_type es igual a 1 en ambas y, o bien delta\_pic\_order\_cnt[ 0 ] tiene valor diferente, o bien delta\_pic\_order\_cnt[ 1 ] tienen valor diferente.
- nal\_unit\_type tiene un valor diferente, y uno de los valores de nal\_unit\_type es igual a 5.
- nal\_unit\_type es igual a 5 en ambas e idr\_pic\_id tienen valores diferentes.

NOTA – Para detectar los límites entre unidades de acceso también es posible utilizar algunas unidades NAL VCL en imágenes codificadas redundantes o algunas unidades NAL distintas de VCL (por ejemplo una unidad NAL delimitadora de unidades de acceso) y por consiguiente pueden servir de ayuda para detectar el inicio de una nueva imagen codificada primaria.

#### **7.4.1.2.5 Orden de unidades NAL VCL y relación con imágenes codificadas**

Cada unidad NAL VCL forma parte de una imagen codificada.

El orden de unidades NAL VCL en una imagen IDR codificada está limitado del modo siguiente.

- Si se permite la ordenación de sectores arbitraria, como se describe en el anexo A, las unidades NAL de sector codificado de una imagen IDR pueden tener cualquier orden.
- De lo contrario (no se permite la ordenación de sectores arbitraria), las unidades NAL de sector codificado de una imagen IDR estarán en orden creciente de dirección de macrobloque contando a partir del primer macrobloque de cada unidad NAL de sector codificado de una imagen IDR.

El orden de las unidades NAL VCL dentro de una imagen que no es IDR codificada está limitado del modo siguiente.

- Si se permite una ordenación de sectores arbitraria, como se especifica en el anexo A, las unidades NAL de sector codificado de una imagen que no es IDR o las unidades NAL de la partición A de datos de sector codificado pueden tener cualquier orden. La unidad NAL de la partición A de datos de sector codificado con un determinado valor de slice\_id estará antes que la unidad NAL de la partición B de datos de sector codificado con el mismo valor de slice\_id. La unidad NAL de la partición A de datos de sector codificado con un determinado valor de slice\_id estará antes que cualquier unidad NAL de la partición C de datos de sector codificado con el mismo valor de slice\_id. Cuando haya una unidad NAL de la partición B de datos de sector codificado, estará antes que la unidad NAL de la partición C de datos de sector codificado con el mismo valor de slice\_id.
- De lo contrario (no se permite una ordenación de sectores arbitraria) las unidades NAL de sector codificado de una imagen que no es IDR o las unidades NAL de la partición A de datos de sector codificado estarán en orden creciente de dirección de macrobloque empezando por el primer macrobloque de cada unidad NAL de sector codificado de una imagen que no es IDR o de cada unidad NAL de la partición A de datos de sector codificado. La unidad NAL de la partición A de datos de sector codificado con un determinado valor de slice\_id estará inmediatamente antes de cualquier unidad NAL de la partición B de datos de sector codificado que haya con el mismo valor de slice\_id. La unidad NAL de la partición A de datos de sector codificado con un determinado valor de slice\_id estará inmediatamente antes que cualquier unidad NAL de la partición C de datos de sector codificado que haya con el mismo valor de slice\_id, cuando no haya ninguna unidad NAL de la partición B de datos de sector codificado con el mismo valor de slice\_id. Cuando haya alguna unidad NAL de la partición B de datos de sector codificado con un determinado valor de slice\_id estará inmediatamente antes que cualquier unidad NAL de la partición C de datos de sector codificado con el mismo valor de slice\_id.

Si bien puede haber unidades NAL cuyo nal\_unit\_type sea igual a 12, éstas no estarán antes que la primera unidad NAL VCL de la imagen codificada primaria dentro de la unidad de acceso.

Si bien puede haber unidades NAL cuyo `nal_unit_type` sea igual a 0 o esté entre 24 y 31, inclusive, que no están especificados, estas unidades no estarán antes que la primera unidad NAL VCL de la imagen codificada primaria dentro de la unidad de acceso.

Las unidades NAL cuyo `nal_unit_type` esté entre 19 y 23, inclusive, los cuales son valores reservados, no estarán antes que la primera unidad NAL VCL de la imagen codificada primaria dentro de una unidad de acceso (cuando se especifique en el futuro por la UIT-T| ISO/CEI).

## 7.4.2 Semántica de las cabidas útiles de secuencia de bits en bruto y de sus bits de cola

### 7.4.2.1 Semántica de la RBSP de conjunto de parámetros secuencia

**profile\_idc** y **level\_idc** indican el perfil y el nivel con los que es conforme el tren de bits, según se especifica en el anexo A.

**constraint\_set0\_flag** igual a 1 indica que el tren de bits cumple todas las restricciones indicadas en la subcláusula A.2.1. **constraint\_set0\_flag** igual a 0 indica que quizá el tren de bits cumpla todas las restricciones especificadas en la subcláusula A.2.1.

**constraint\_set1\_flag** igual a 1 indica que el tren de bits cumple todas las restricciones especificadas en la subcláusula A.2.2. **constraint\_set1\_flag** igual a 0 indica que quizá el tren de bits cumple todas las restricciones especificadas en la subcláusula A.2.2.

**constraint\_set2\_flag** igual a 1 indica que el tren de bits cumple todas las restricciones especificadas en la subcláusula A.2.3. **constraint\_set2\_flag** igual a 0 indica que quizá el tren de bits cumpla todas las restricciones especificadas en la subcláusula A.2.3.

NOTA – Cuando más de uno de los `constraint_set0_flag`, `constraint_set1_flag`, o `constraint_set2_flag` sea igual a 1, el flujo de bits cumple las restricciones de todas las subcláusulas indicadas de la subcláusula A.2.

**reserved\_zero\_5bits** será igual a 0 en los trenes de bits conformes con esta Recomendación | Norma Internacional. En el futuro, el UIT-T | ISO/CEI podrá especificar otros valores de `reserved_zero_5bits`. Los decodificadores deberán hacer caso omiso del valor de `reserved_zero_5bits`.

**seq\_parameter\_set\_id** identifica el conjunto de parámetros secuencia al cual hace referencia el conjunto de parámetros imagen. El valor de `seq_parameter_set_id` estará entre 0 y 31, inclusive.

NOTA – En la medida de lo posible, los codificadores deberían utilizar valores distintos de `seq_parameter_set_id` cuando los valores de otros elementos sintácticos de conjunto de parámetros secuencia sean diferentes, en lugar de cambiar los valores de los elementos sintácticos correspondientes a un determinado valor de `seq_parameter_set_id`.

**log2\_max\_frame\_num\_minus4** especifica el valor de la variable `MaxFrameNum` que se utiliza en los cálculos en los que interviene `frame_num` del modo siguiente:

$$\text{MaxFrameNum} = 2^{(\text{log2\_max\_frame\_num\_minus4} + 4)} \quad (7-1)$$

El valor de `log2_max_frame_num_minus4` estará entre 0 y 12, inclusive.

**pic\_order\_cnt\_type** especifica el método para decodificar el número de orden de la imagen (según se especifica en la subcláusula 8.2.1). El valor `pic_order_cnt_type` estará entre 0 y 2, inclusive.

`pic_order_cnt_type` será distinto de 2 en las secuencias de vídeo codificado que contengan una de las siguientes posibilidades:

- una unidad de acceso que contiene un cuadro que no es de referencia seguida inmediatamente por una unidad de acceso que contiene una imagen que no es de referencia;
- dos unidades de acceso, cada una con un campo, de modo que los dos campos formen un par de campos que no son de referencias complementarios seguidas inmediatamente por una unidad de acceso que contiene una imagen que no es de referencia;

- una unidad de acceso que contiene un campo que no es de referencia seguido inmediatamente por una unidad de acceso que contiene otra imagen que no es de referencia y que no forma un par de campos que no son de referencia complementarios con la primera de las dos unidades de acceso.

**log2\_max\_pic\_order\_cnt\_lsb\_minus4** especifica el valor de la variable `MaxPicOrderCntLsb` que se utiliza en el proceso de decodificación del número de orden de la imagen, según se describe en la subcláusula 8.2.1:

$$\text{MaxPicOrderCntLsb} = 2^{(\text{log2\_max\_pic\_order\_cnt\_lsb\_minus4} + 4)} \quad (7-2)$$

El valor de `log2_max_pic_order_cnt_lsb_minus4` estará entre 0 y 12, inclusive.

**delta\_pic\_order\_always\_zero\_flag** igual a 1 indica que no hay `delta_pic_order_cnt[ 0 ]` y `delta_pic_order_cnt[ 1 ]` en las cabeceras de sector de la secuencia y se considerará que son iguales a 0. `delta_pic_order_always_zero_flag` igual a 0 indica que hay `delta_pic_order_cnt[ 0 ]` en las cabeceras de sector de la secuencia y que puede haber `delta_pic_order_cnt[ 1 ]` en las cabeceras de sector de la secuencia.

**offset\_for\_non\_ref\_pic** se utiliza para calcular el número de orden de una imagen que no es de referencia, según se describe en 8.2.1. El valor de `offset_for_non_ref_pic` estará entre  $-2^{31}$  y  $2^{31} - 1$ , inclusive.

**offset\_for\_top\_to\_bottom\_field** se utiliza para calcular el número de orden de la imagen del campo inferior de un cuadro, según se describe en 8.2.1. El valor de `offset_for_top_to_bottom_field` estará entre  $-2^{31}$  y  $2^{31} - 1$ , inclusive.

**num\_ref\_frames\_in\_pic\_order\_cnt\_cycle** se utiliza en el proceso de decodificación del número de orden de la imagen, según se especifica en la subcláusula 8.2.1. El valor de `num_ref_frames_in_pic_order_cnt_cycle` estará entre 0 y 255, inclusive.

**offset\_for\_ref\_frame[ i ]** es un elemento de la lista de valores `num_ref_frames_in_pic_order_cnt_cycle` utilizado en el proceso de decodificación del número de orden de la imagen, según se describe en la subcláusula 8.2.1. El valor de `offset_for_ref_frame[ i ]` estará entre  $-2^{31}$  y  $2^{31} - 1$ , inclusive.

**num\_ref\_frames** especifica el número máximo de cuadros de referencia de corto y largo alcance, de pares de campo de referencia complementarios y de campos de referencia no apareados que puede utilizar el proceso de decodificación para la predicción inter de imágenes en la secuencia. `num_ref_frames` también determina el tamaño de la ventana de deslizamiento, según se describe en la subcláusula 8.2.5.3. El valor de `num_ref_frames` estará entre 0 y `MaxDpbSize` (tal y como se especifica en la subcláusula A.3.1), inclusive.

**gaps\_in\_frame\_num\_value\_allowed\_flag** especifica los valores permitidos de `frame_num`, según se describe en la subcláusula 7.4.3 y el proceso de decodificación en caso de que se descubra una diferencia entre los valores de `frame_num`, según se describe en la subcláusula 8.2.5.2.

**pic\_width\_in\_mbs\_minus1** más 1 especifica la anchura de cada imagen decodificada en unidades de macrobloque.

La variable anchura de la imagen en unidades de macrobloque se calcula del modo siguiente:

$$\text{PicWidthInMbs} = \text{pic\_width\_in\_mbs\_minus1} + 1 \quad (7-3)$$

La variable anchura de la imagen correspondiente a la componente luma se calcula del modo siguiente:

$$\text{PicWidthInSamples}_L = \text{PicWidthInMbs} * 16 \quad (7-4)$$

La variable anchura de la imagen correspondiente a las componentes croma se calcula del modo siguiente:

$$\text{PicWidthInSamples}_C = \text{PicWidthInMbs} * 8 \quad (7-5)$$

**pic\_height\_in\_map\_units\_minus1** más 1 especifica la altura de un cuadro o campo decodificado en unidades de mapeado de grupo de sector.

Las variables **PicHeightInMapUnits** y **PicSizeInMapUnits** se calculan del modo siguiente:

$$\text{PicHeightInMapUnits} = \text{pic\_height\_in\_map\_units\_minus1} + 1 \quad (7-6)$$

$$\text{PicSizeInMapUnits} = \text{PicWidthInMbs} * \text{PicHeightInMapUnits} \quad (7-7)$$

**frame\_mbs\_only\_flag** igual a 0 indica que las imágenes codificadas en la secuencia de vídeo codificado pueden ser campos codificados o cuadros codificados. **frame\_mbs\_only\_flag** igual a 1 indica que cada imagen codificada de la secuencia de vídeo codificado es un cuadro codificado que contiene sólo macrobloques cuadro.

La gama de valores permitidos de **pic\_width\_in\_mbs\_minus1**, **pic\_height\_in\_map\_units\_minus1**, y **frame\_mbs\_only\_flag** se especifica mediante las restricciones indicadas en el anexo A.

En función de **frame\_mbs\_only\_flag**, la semántica que se asigna a **pic\_height\_in\_map\_units\_minus1** es diferente.

- Si **frame\_mbs\_only\_flag** es igual a 0, **pic\_height\_in\_map\_units\_minus1** más 1 es la altura de un campo en unidades de macrobloque.
- De lo contrario (**frame\_mbs\_only\_flag** es igual a 1), **pic\_height\_in\_map\_units\_minus1** más 1 es la altura de un cuadro en unidades de macrobloque.

La variable **FrameHeightInMbs** se calcula del modo siguiente:

$$\text{FrameHeightInMbs} = (2 - \text{frame\_mbs\_only\_flag}) * \text{PicHeightInMapUnits} \quad (7-8)$$

**mb\_adaptive\_frame\_field\_flag** igual a 0 indica que se intercambian macrobloques cuadro y campo dentro de la imagen. **mb\_adaptive\_frame\_field\_flag** igual a 1 indica que es posible intercambiar macrobloques cuadro y campo dentro de cuadros. Cuando no haya **mb\_adaptive\_frame\_field\_flag**, se sobreentenderá que es igual a 0.

**direct\_8x8\_inference\_flag** especifica el método utilizado en el proceso de cálculo de vectores de movimiento luma para **B\_Skip**, **B\_Direct\_16x16** y **B\_Direct\_8x8**, según se describe en la subcláusula 8.4.1.2. Cuando **frame\_mbs\_only\_flag** sea igual a 0, **direct\_8x8\_inference\_flag** será igual a 1.

**frame\_cropping\_flag** igual a 1 indica que los parámetros de desplazamiento de recorte de cuadro aparecen a continuación en el conjunto de parámetros secuencia. **frame\_cropping\_flag** igual a 0 indica que no hay parámetros de traslación de recorte de cuadro.

**frame\_crop\_left\_offset**, **frame\_crop\_right\_offset**, **frame\_crop\_top\_offset**, **frame\_crop\_bottom\_offset** indican las muestras de un cuadro dentro de un rectángulo, del modo siguiente.

- Si **frame\_mbs\_only\_flag** es igual a 1, el rectángulo de recorte contiene las muestras luma con coordenadas horizontales entre  $2 * \text{frame\_crop\_left\_offset}$  y  $\text{PicWidthInSamples}_L - (2 * \text{frame\_crop\_right\_offset} + 1)$  y coordenadas verticales entre  $2 * \text{frame\_crop\_top\_offset}$  y  $(\text{FrameHeightInMbs} * 16) - (2 * \text{frame\_crop\_bottom\_offset} + 1)$ , inclusive. En este caso el valor de **frame\_crop\_left\_offset** estará entre 0 y  $8 * \text{PicWidthInMbs} - (\text{frame\_crop\_right\_offset} + 1)$ , inclusive; y el valor de **frame\_crop\_top\_offset** estará entre 0 y  $8 * \text{FrameHeightInMbs} - (\text{frame\_crop\_bottom\_offset} + 1)$ , inclusive.
- De lo contrario (**frame\_mbs\_only\_flag** es igual a 0), el rectángulo de recorte contiene las muestras luma con coordenadas horizontales entre  $2 * \text{frame\_crop\_left\_offset}$  y  $\text{PicWidthInSamples}_L - (2 * \text{frame\_crop\_right\_offset} + 1)$  y coordenadas verticales entre  $4 * \text{frame\_crop\_top\_offset}$  y  $(\text{FrameHeightInMbs} * 16) - (4 * \text{frame\_crop\_bottom\_offset} + 1)$ , inclusive. En este caso el valor de **frame\_crop\_left\_offset** estará entre 0

y  $8 * \text{PicWidthInMbs} - (\text{frame\_crop\_right\_offset} + 1)$ , inclusive; y el valor de  $\text{frame\_crop\_top\_offset}$  entre 0 y  $4 * \text{FrameHeightInMbs} - (\text{frame\_crop\_bottom\_offset} + 1)$ , inclusive.

Cuando `frame_cropping_flag` sea igual a 0, se sobreentenderán los siguientes valores: `frame_crop_left_offset = 0`, `frame_crop_right_offset = 0`, `frame_crop_top_offset = 0`, y `frame_crop_bottom_offset = 0`.

Las muestras especificadas de las dos matrices croma son las muestras cuyas coordenadas del cuadro son  $(x / 2, y / 2)$ , siendo  $(x, y)$  las coordenadas del cuadro de las muestras luma especificadas.

En los campos decodificados, las muestras especificadas del campo decodificado son las muestras que están en el rectángulo especificado en las coordenadas del cuadro.

`vui_parameters_present_flag` igual a 1 indica que lo que aparece a continuación en el tren de bits es la estructura sintáctica de `vui_parameters()` especificada en el anexo E. `vui_parameters_present_flag` igual a 0 indica que lo que aparece a continuación en el tren de bits no es la estructura de sintaxis de `vui_parameters()` especificada en el anexo E.

#### 7.4.2.2 Semántica de la RBSP de conjunto de parámetros imagen

`pic_parameter_set_id` indica el conjunto de parámetros imagen al que se hace referencia en la cabecera de sector. El valor de `pic_parameter_set_id` estará entre 0 y 255, inclusive.

`seq_parameter_set_id` hace referencia al conjunto de parámetros secuencia activo. El valor de `seq_parameter_set_id` estará entre 0 y 31, inclusive.

`entropy_coding_mode_flag` indica el método de decodificación entrópica que se aplicará a los elementos sintácticos para los cuales aparecen dos descriptores en los cuadros de sintaxis.

- Si `entropy_coding_mode_flag` es igual a 0, se aplica el método especificado por el descriptor izquierdo en el cuadro de sintaxis (codificación Exp-Golomb, véase subcláusula 9.1 o CAVLC, véase subcláusula 9.2).
- De lo contrario (`entropy_coding_mode_flag` es igual a 1), se aplica el método especificado en el descriptor derecho en el cuadro de sintaxis (CABAC, véase subcláusula 9.3).

`pic_order_present_flag` igual a 1 indica que las cabeceras de sector contienen los elementos sintácticos relacionados con el número de orden de la imagen, según se especifica en la subcláusula 7.3.3. `pic_order_present_flag` igual a 0 indica que las cabeceras de sector no contienen los elementos sintácticos relacionados con el número de orden de la imagen.

`num_slice_groups_minus1` más 1 especifica el número de grupos de sector de la imagen. Cuando `num_slice_groups_minus1` es igual a 0, todos los sectores de la imagen pertenecen al mismo grupo de sector. Los valores permitidos de `num_slice_groups_minus1` se especifica en el anexo A.

`slice_group_map_type` especifica cómo se codifica el mapeado de unidades de mapeado de grupos de sector en grupos de sector. El valor de `slice_group_map_type` estará entre 0 y 6, inclusive.

`slice_group_map_type` igual a 0 indica grupos de sectores intercalados.

`slice_group_map_type` igual a 1 indica un mapeado de grupos de sectores dispersos.

`slice_group_map_type` igual a 2 indica que hay al menos un grupo de sectores "de primer plano" y un grupo de sectores "sobrantes"

Los valores de `slice_group_map_type` igual a 3, 4 y 5 indican grupo de sectores cambiantes. Cuando `num_slice_groups_minus1` sea distinto de 1, `slice_group_map_type` será distinto a 3, 4 ó 5.

`slice_group_map_type` igual a 6 indica una asignación explícita de un grupo de sectores a cada unidad de mapeado de grupos de sectores.



Las unidades de mapeado de grupos de sectores se definen del modo siguiente:

- Si `frame_mbs_only_flag` es igual a 0 y `mb_adaptive_frame_field_flag` es igual a 1 y la imagen codificada es un cuadro, las unidades de mapeado de grupos de sector son unidades de par de macrobloques.
- De lo contrario, si `frame_mbs_only_flag` es igual a 1 o la imagen codificada es un campo, las unidades de mapeado de grupos de sectores son unidades de macrobloques.
- De lo contrario (`frame_mbs_only_flag` es igual a 0 y `mb_adaptive_frame_field_flag` es igual a 0 y la imagen codificada es un cuadro), las unidades de mapeado de grupos de sectores son unidades de dos macrobloques que están contiguos verticalmente como en un par de macrobloques de cuadro de un cuadro MBAFF.

`run_length_minus1[ i ]` se utiliza para especificar el número de unidades de mapeado de grupos de sectores consecutivas que se asignan al *i*-ésimo grupo de sectores en el orden de barrido por filas de unidades de mapeado de grupos de sectores. El valor de `run_length_minus1[ i ]` estará entre 0 y `PicSizeInMapUnits - 1`, inclusive.

`top_left[ i ]` y `bottom_right[ i ]` especifican las esquinas superior izquierda e inferior derecha de un rectángulo, respectivamente. `top_left[ i ]` y `bottom_right[ i ]` son posiciones de unidades de mapeado de grupos de sectores en el orden de barrido por filas de la imagen correspondiente a unidades de mapeado de grupos de sectores. Para cada rectángulo *i*, los valores de los elementos sintácticos `top_left[ i ]` y `bottom_right[ i ]` cumplirán todas las restricciones siguientes.

- `top_left[ i ]` será menor o igual que `bottom_right[ i ]`, y `bottom_right[ i ]` será menor que `PicSizeInMapUnits`.
- $(\text{top\_left}[ i ] \% \text{PicWidthInMbs})$  será menor o igual al valor de  $(\text{bottom\_right}[ i ] \% \text{PicWidthInMbs})$ .

`slice_group_change_direction_flag` se utiliza junto con `slice_group_map_type` para especificar el tipo de mapeado concreto cuando `slice_group_map_type` es igual a 3, 4 ó 5.

`slice_group_change_rate_minus1` se utiliza para especificar la variable `SliceGroupChangeRate`. `SliceGroupChangeRate` especifica el múltiplo del número de unidades de mapeado de grupos de sector por el cual se puede cambiar el tamaño de un grupo de sectores de una imagen a la siguiente. El valor de `slice_group_change_rate_minus1` estará entre 0 y `PicSizeInMapUnits - 1`, inclusive. La variable `SliceGroupChangeRate` se define del modo siguiente:

$$\text{SliceGroupChangeRate} = \text{slice\_group\_change\_rate\_minus1} + 1 \quad (7-9)$$

`pic_size_in_map_units_minus1` se utiliza para especificar el número de unidades de mapeado de grupos de sectores en la imagen. `pic_size_in_map_units_minus1` será igual a `PicSizeInMapUnits - 1`.

`slice_group_id[ i ]` determina el grupo de sector de la *i*-ésima unidad de mapeado de grupos de sector en el orden barrido por filas. El tamaño del elemento sintáctico `slice_group_id[ i ]` es  $\text{Ceil}(\text{Log}_2(\text{num\_slice\_groups\_minus1} + 1))$  bits. El valor de `slice_group_id[ i ]` estará entre 0 y `num_slice_groups_minus1`, inclusive.

`num_ref_idx_l0_active_minus1` especifica el valor máximo del índice de referencia de la lista 0 de imágenes de referencia que se utilizará para decodificar cada sector de la imagen en el que se utiliza la lista 0 cuando `num_ref_idx_active_override_flag` es igual a 0 para el sector. Cuando `MbaffFrameFlag` es igual a 1, `num_ref_idx_l0_active_minus1` es el valor máximo del índice para la decodificación de macrobloques cuadro y  $2 * \text{num\_ref\_idx\_l0\_active\_minus1} + 1$  es el valor máximo del índice para la decodificación de macrobloques campo. El valor de `num_ref_idx_l0_active_minus1` estará entre 0 y 31, inclusive.

`num_ref_idx_l1_active_minus1` tiene la misma semántica que `num_ref_idx_l0_active_minus1` sustituyendo l0 y lista 0 por l1 y lista 1, respectivamente.

**weighted\_pred\_flag** igual a 0 indica que no se aplica la predicción ponderada a los sectores P y SP. **weighted\_pred\_flag** igual a 1 indica que se aplica la predicción ponderada a los sectores P y SP.

**weighted\_bipred\_idc** igual a 0 indica que se aplica la predicción ponderada por defecto a los sectores B. **weighted\_bipred\_idc** igual a 1 indica que se aplica la predicción ponderada explícita a los sectores B. **weighted\_bipred\_idc** igual a 2 indica que se aplica la predicción ponderada implícita a los sectores B. El valor de **weighted\_bipred\_idc** estará entre 0 y 2, inclusive.

**pic\_init\_qp\_minus26** especifica el valor inicial menos 26 de  $SliceQP_Y$  para cada sector. El valor inicial se modifica en la capa sector cuando se decodifica un valor distinto de cero de **slice\_qp\_delta**, y se modifica nuevamente cuando al decodificar un valor distinto de cero de **mb\_qp\_delta** en la capa macrobloque. El valor de **pic\_init\_qp\_minus26** estará entre -26 y +25, inclusive.

**pic\_init\_qs\_minus26** especifica el valor inicial menos 26 de  $SliceQS_Y$  para todos los macrobloques en los sectores SP o SI. El valor inicial se modifica en la capa sector cuando se decodifica un valor distinto de cero de **slice\_qs\_delta**. El valor de **pic\_init\_qs\_minus26** estará entre -26 y +25, inclusive.

**chroma\_qp\_index\_offset** especifica la traslación que se deberá sumar a  $QP_Y$  y  $QS_Y$  para el direccionamiento del cuadro de valores  $QP_C$ . El valor de **chroma\_qp\_index\_offset** estará entre -12 y +12, inclusive.

**deblocking\_filter\_control\_present\_flag** igual a 1 indica que la cabecera de sector contiene el conjunto de elementos sintácticos que controlan las características del filtrado de bloques. **deblocking\_filter\_control\_present\_flag** igual a 0 indica que las cabeceras del sector no contienen el conjunto de elementos sintácticos que controlan las características del filtrado de bloques y se aplican los valores por defecto.

**constrained\_intra\_pred\_flag** igual a 0 indica que en la predicción inter es posible utilizar datos residuales y muestras decodificadas de macrobloques adyacentes codificados utilizando los modos de predicción de macrobloque inter para predecir macrobloques codificados utilizando modos de predicción macrobloque intra. **constrained\_intra\_pred\_flag** igual a 1 indica predicción intra restringida, en la que para predecir macrobloques codificados utilizando modos de predicción macrobloque Intra sólo se utilizan datos residuales y muestras decodificadas a partir de macrobloques de tipo I o SI.

**redundant\_pic\_cnt\_present\_flag** igual a 0 indica que el elemento sintáctico **redundant\_pic\_cnt** no está contenido en cabeceras de sector, particiones B de datos, y particiones C de datos que hacen referencia (bien directamente o por asociación con la correspondiente partición A de datos) al conjunto de parámetros imagen. **redundant\_pic\_cnt\_present\_flag** igual a 1 indica que el elemento sintáctico **redundant\_pic\_cnt** está presente en todas las cabeceras de sector, particiones B de datos, y particiones C de datos que hacen referencia (directamente o por asociación con la correspondiente partición A de datos) al conjunto de parámetros imagen.

### 7.4.2.3 Semántica de la RBSP de información de perfeccionamiento complementaria

La información de perfeccionamiento complementaria (SEI) contiene información que no es necesaria para decodificar las muestras de imágenes codificadas contenidas en unidades NAL VCL.

#### 7.4.2.3.1 Semántica de mensajes de información de perfeccionamiento complementario

Cada unidad NAL SEI contiene al menos un mensaje SEI. Cada mensaje SEI consta de las variables que especifican el tipo **payloadType** y el tamaño **payloadSize** de la cabida útil SEI. Las cabidas útiles SEI se especifican en el anexo D. El tamaño de la cabida útil SEI calculada **payloadSize** se especifica en bytes y será igual al número de bytes de la cabida útil SEI.

**ff\_byte** es un byte igual a 0xFF que indica la necesidad de una representación más amplia de la estructura sintáctica que se utiliza en ella.

**last\_payload\_type\_byte** es el último byte del tipo de cabida útil de un mensaje SEI.

**last\_payload\_size\_byte** es el último byte del tamaño de un mensaje SEI.

#### 7.4.2.4 Semántica de la RBSP del delimitador de unidad de acceso

Es posible utilizar el delimitador de unidad de acceso para indicar el tipo de sectores que contiene una imagen codificada primaria y para simplificar la detección de los límites entre unidades de acceso. No existe un proceso de decodificación normativo relacionado con el delimitador de unidades de acceso.

**primary\_pic\_type** indica que los valores **slice\_type** para todos los sectores de la imagen codificada primaria forman parte del conjunto enumerado en el cuadro 7-2 para un determinado valor de **primary\_pic\_type**.

**Cuadro 7-2 – Significado de primary\_pic\_type**

<b>primary_pic_type</b>	<b>valores de slice_type que pueden contener la imagen codificada primaria</b>
0	I
1	I, P
2	I, P, B
3	SI
4	SI, SP
5	I, SI
6	I, SI, P, SP
7	I, SI, P, SP, B

#### 7.4.2.5 Semántica de la RBSP fin de secuencia

La RBSP fin de secuencia especifica que la siguiente unidad de acceso en el tren de bits, en orden de decodificación (si lo hubiere) es una unidad de acceso IDR. El contenido sintáctico de la SODB y la RBSP para la RBSP fin de secuencia está vacío. El proceso de decodificación normativo para las RBSP fin de secuencia no está especificado.

#### 7.4.2.6 Semántica de la RBSP fin de tren

La RBSP fin de tren indica que no hay más unidades NAL en el tren de bits después de la RBSP fin de tren en el orden de codificación. El contenido sintáctico de la SODB y la RBSP correspondiente a la RBSP fin de tren está vacío. El proceso de codificación normativo para la RBSP fin de tren no está especificado.

#### 7.4.2.7 Semántica de la RBSP datos de relleno

La RBSP datos de relleno contiene bytes cuyo valor será igual a 0xFF. El proceso de codificación normativo para la RBSP datos de relleno no está especificado.

**ff\_byte** es un byte de valor 0xFF.

#### 7.4.2.8 Semántica de la RBSP capa de sector sin particiones

La RBSP capa de sector sin particiones consta de una cabecera de sector y de datos de sector.

## 7.4.2.9 Semántica de la RBSP particiones de datos de sector

### 7.4.2.9.1 Semántica de la RBSP de la partición A de datos de sector

Cuando se utilizan particiones de datos de sector, los datos codificados de un mismo sector se subdividen en tres particiones distintas. La partición A contiene todos los elementos sintácticos de la categoría 2.

Los elementos sintácticos de la categoría 2 son todos los elementos sintácticos en la cabecera de sector y las estructuras sintácticas datos de sector distintas de los elementos sintácticos de la estructura sintáctica residual().

**slice\_id** indica el sector relacionado con la partición de datos. Cada sector tendrá un valor único de **slice\_id** dentro de la imagen codificada que contiene el sector. Cuando no se permite la ordenación de sectores arbitraria, que se describe en el anexo A, el primer sector de una imagen codificada en orden de decodificación, tendrá **slice\_id** igual a 0 y el valor de **slice\_id** se incrementará en una unidad para cada sector subsiguiente de la imagen codificada, en orden de decodificación.

Los valores posibles de **slice\_id** son los siguientes:

- Si **MbaffFrameFlag** es igual a 0, **slice\_id** tendrá un valor entre 0 y  $\text{PicSizeInMbs} - 1$ , inclusive.
- De lo contrario (**MbaffFrameFlag** es igual a 1), **slice\_id** tendrá un valor entre 0 y  $\text{PicSizeInMbs} / 2 - 1$ , inclusive.

### 7.4.2.9.2 Semántica de la RBSP partición B de datos de sector

Cuando se utilizan particiones de datos de sector, los datos codificados correspondientes a un mismo sector se dividen en de una a tres particiones diferentes. La partición B de datos de sector contiene todos los elementos sintácticos de la categoría 3.

Los elementos sintácticos de la categoría 3 son todos los elementos sintácticos de la estructura sintáctica residual() y de las estructuras sintácticas que se utilizan dentro de esa estructura sintáctica para macrobloques genéricos de tipo I y SI, según se especifica en el cuadro 7-7.

**slice\_id** tiene la misma semántica que la especificada en la subcláusula 7.4.2.9.1

**redundant\_pic\_cnt** será igual a 0 para los sectores y las particiones de datos de sector que pertenezcan a la imagen codificada primaria. **redundant\_pic\_cnt** será mayor que 0 para los sectores codificados y las particiones de datos de sector codificados en imágenes codificadas redundantes. Si no hubiere **redundant\_pic\_cnt**, se supondrá que su valor es 0. El valor de **redundant\_pic\_cnt** estará entre 0 y 127, inclusive.

La presencia de una RBSP partición B de datos de sector se especifica del modo siguiente:

- Si los elementos sintácticos de la RBSP partición A de datos de sector indican la presencia de elementos sintácticos de la categoría 3 en los datos de sector correspondientes a un sector, habrá una RBSP partición B de datos de sector cuyos valores de **slice\_id** y de **redundant\_pic\_cnt** serán los mismos que en la RBSP partición A de datos de sector.
- De lo contrario (los elementos sintácticos de una RBSP partición A de datos de sector no indican la presencia de elementos sintácticos de la categoría 3 en los datos de sector correspondientes al sector), no habrá RBSP partición B de datos de sector que tenga el mismo valor de **slice\_id** y de **redundant\_pic\_cnt** que la RBSP partición A de datos de sector.

### 7.4.2.9.3 Semántica de la RBSP partición C de datos de sector

Cuando se utilizan particiones de datos de sector, los datos codificados correspondientes a un mismo sector se dividen en tres particiones distintas. La partición C de datos de sector contiene todos los elementos sintácticos de la categoría 4.

Los elementos sintácticos de la categoría 4 son todos elementos sintácticos de la estructura sintáctica residual() y de las estructuras sintácticas utilizadas en esa estructura sintáctica para macrobloques genéricos de tipo P y B, según se especifica en el cuadro 7-7.

**slice\_id** tiene la misma semántica que la especificada en la subcláusula 7.4.2.9.1.

**redundant\_pic\_cnt** tiene la misma semántica que la especificada en la subcláusula 7.4.2.9.2.

La presencia de una RBSP partición C de datos de sector se especifica del modo siguiente:

- Si los elementos sintácticos de la RBSP partición A de datos de sector indican la presencia de elementos sintácticos de la categoría 4 en los datos de sector correspondientes a un sector, habrá una RBSP partición C de datos de sector cuyos valores de **slice\_id** y de **redundant\_pic\_cnt** serán los mismos que en la RBSP partición A de datos de sector.
- De lo contrario (los elementos sintácticos de una RBSP de partición A de datos de sector no indican la presencia de elementos sintácticos de la categoría 4 en los datos de sector correspondientes al sector), no habrá RBSP partición C de datos de sector que tenga el mismo valor de **slice\_id** y de **redundant\_pic\_cnt** que la RBSP partición A de datos de sector.

### 7.4.2.10 Semántica de los bits de cola del sector de la RBSP

**cabac\_zero\_word** es una secuencia de dos bytes alineados por byte y de valor 0x0000.

Sea **NumBytesInVclNALunits** la suma de los valores de **NumBytesInNALunit** de todas las unidades NAL VCL de una imagen codificada.

Sea **BinCountsInNALunits** el número de veces en que se invoca la función de proceso analítico **DecodeBin()**, que se especifica en la subcláusula 9.3.3.2, con el fin de decodificar los contenidos de todas las unidades NAL VCL de una imagen codificada. Si **entropy\_coding\_mode\_flag** es igual a 1, **BinCountsInNALunits** no será mayor que  $(32 \div 3) * \text{NumBytesInVclNALunits} + 96 * \text{PicSizeInMbs}$ .

NOTA – Se puede cumplir la restricción del número máximo de bins que resulta de decodificar el contenido de unidades NAL de capa sector mediante la inserción de una serie de elementos sintácticos **cabac\_zero\_word** para aumentar el valor de **NumBytesInVclNALunits**. Cada **cabac\_zero\_word** se representa en una unidad NAL mediante la secuencia de tres bytes 0x000003 (debido a las restricciones que se aplican a los contenidos de unidades NAL y que obligan a incluir un **emulation\_prevention\_three\_byte** para cada **cabac\_zero\_word**).

### 7.4.2.11 Semántica de bits de cola de la RBSP

**rbsp\_stop\_one\_bit** es un solo bit de valor 1.

**rbsp\_alignment\_zero\_bit** es un solo bit de valor 0.

## 7.4.3 Semántica de cabeceras de sector

Si los hubiere, el valor de los elementos sintácticos cabecera de sector **pic\_parameter\_set\_id**, **frame\_num**, **field\_pic\_flag**, **bottom\_field\_flag**, **idr\_pic\_id**, **pic\_order\_cnt\_lsb**, **delta\_pic\_order\_cnt\_bottom**, **delta\_pic\_order\_cnt[ 0 ]**, **delta\_pic\_order\_cnt[ 1 ]**, **sp\_for\_switch\_flag**, y **slice\_group\_change\_cycle** será el mismo en todas las cabeceras de sector de una imagen codificada.

**first\_mb\_in\_slice** especifica la dirección del primer macrobloque en el sector. Cuando no se permite la ordenación de sectores arbitraria, que se describe en el anexo A, el valor de

first\_mb\_in\_slice no será inferior al valor de first\_mb\_in\_slice para los demás sectores de la imagen considerada que precece al sector considerado en el orden de decodificación.

La dirección del primer macrobloque del sector se calcula del modo siguiente:

- Si MbaffFrameFlag es igual a 0, first\_mb\_in\_slice es la dirección del macrobloque del primer macrobloque en el sector, y el valor de first\_mb\_in\_slice estará entre 0 y PicSizeInMbs – 1, inclusive.
- De lo contrario (MbaffFrameFlag es igual a 1), first\_mb\_in\_slice \* 2 es la dirección del macrobloque del primer macrobloque en el sector, que es el macrobloque superior del primer par de macrobloques en el sector, y el valor first\_mb\_in\_slice estará entre 0 y PicSizeInMbs / 2 – 1, inclusive.

**slice\_type** especifica el tipo de codificación del sector de conformidad con el cuadro 7-3.

**Cuadro 7-3 – Nombres correspondientes a slice\_type**

slice_type	Nombre de slice_type
0	P (Sector P)
1	B (Sector B)
2	I (Sector I)
3	SP (Sector SP)
4	SI (Sector SI)
5	P (Sector P)
6	B (Sector B)
7	I (Sector I)
8	SP (Sector SP)
9	SI (Sector SI)

Los valores de slice\_type en la gama 5..9 especifican, además del tipo de codificación del sector considerado, que todos los demás sectores de la imagen codificada en cuestión tendrán un valor de slice\_type igual al valor actual de slice\_type o igual al valor actual de slice\_type – 5.

Cuando nal\_unit\_type es igual a 5 (imagen IDR), slice\_type será igual a 2, 4, 7, ó 9.

Cuando num\_ref\_frames es igual a 0, slice\_type será igual a 2, 4, 7 ó 9.

**pic\_parameter\_set\_id** especifica el conjunto de parámetros de imagen que se está utilizando. El valor de pic\_parameter\_set\_id estará entre 0 y 255, inclusive.

**frame\_num** se utiliza como identificador de imágenes y se representará mediante  $\log_2 \text{max\_frame\_num\_minus4} + 4$  bits en el tren de bits. Las instrucciones aplicables a frame\_num son las siguientes:

La variable PrevRefFrameNum se calcula del modo siguiente:

- Si la imagen considerada es una imagen IDR, PrevRefFrameNum es igual a 0.
- De lo contrario (la imagen considerada no es una imagen IDR), PrevRefFrameNum es igual al valor de frame\_num de la unidad de acceso previa en orden de codificación que contiene una imagen de referencia.

El valor de `frame_num` se limita a lo siguiente:

- Si la imagen considerada es una imagen IDR, `frame_num` será igual a 0.
- De lo contrario (la imagen considerada no es IDR), donde por imagen de referencia precedente se entiende a la imagen codificada primaria en la unidad de acceso previa en orden de codificación que contiene una imagen de referencia, el valor de `frame_num` de la imagen considerada no será igual a `PrevRefFrameNum` a no ser que se cumplan todas y cada una de las tres condiciones siguientes:
  - la imagen considerada y la imagen de referencia precedente pertenecen a unidades de acceso consecutivas en orden de decodificación;
  - la imagen considerada y la imagen de referencia precedente son campos de referencia con paridad opuesta;
  - se cumplen al menos una de las siguientes condiciones:
    - la imagen de referencia precedente es una imagen IDR;
    - la imagen de referencia precedente incluye un elemento sintáctico `memory_management_control_operation` de valor igual a 5;  
NOTA – Cuando la imagen de referencia precedente incluye un elemento sintáctico `memory_management_control_operation` cuyo valor es 5, `PrevRefFrameNum` es igual a 0.
    - hay una imagen codificada primaria que precede a la imagen de referencia precedente y la imagen codificada primaria que precede a la imagen de referencia precedente no tiene `frame_num` igual a `PrevRefFrameNum`;
    - hay una imagen codificada primaria que precede a la imagen de referencia precedente y la imagen codificada primaria que precede a la imagen de referencia precedente no es una imagen de referencia.

Cuando el valor de `frame_num` es distinto a `PrevRefFrameNum`, se aplica lo siguiente:

- No habrá campo o cuadro previo alguno, en orden de codificación, que esté marcado como "utilizado para referencia de corto alcance" y que tenga un valor de `frame_num` igual al valor que toma la variable `UnusedShortTermFrameNum` en la siguiente expresión:

```
UnusedShortTermFrameNum = ( PrevRefFrameNum + 1 ) % MaxFrameNum
while( UnusedShortTermFrameNum != frame_num )
    UnusedShortTermFrameNum = ( UnusedShortTermFrameNum + 1 ) % MaxFrameNum
```

(7-10)

- El valor de `frame_num` debe cumplir con lo siguiente.
  - Si `gaps_in_frame_num_value_allowed_flag` es igual a 0, el valor de `frame_num` para la imagen actual debe ser igual a  $( \text{PrevRefFrameNum} + 1 ) \% \text{MaxFrameNum}$ .
  - De lo contrario (`gaps_in_frame_num_value_allowed_flag` es igual a 1), se aplica lo siguiente.
    - Si `frame_num` es mayor que `PrevRefFrameNum`, no habrá imágenes que no sean de referencia en el tren de bits comprendido entre la imagen de referencia anterior y la imagen actual en el orden de decodificación en el que una de las condiciones siguientes es verdadera.
      - El valor de `frame_num` para la imagen que no es de referencia es menor que `PrevRefFrameNum`.
      - El valor de `frame_num` para la imagen que no es de referencia es mayor que el valor de `frame_num` para la imagen actual.
    - De lo contrario (`frame_num` es menor que `PrevRefFrameNum`), no habrá imágenes que no sean de referencia en el tren de bits comprendido entre la imagen de

referencia anterior y la imagen actual en el orden de decodificación en el que las dos condiciones siguientes son verdaderas.

- El valor de `frame_num` para la imagen que no es de referencia es menor que `PrevRefFrameNum`.
- El valor de `frame_num` para la imagen que no es de referencia es mayor que el valor de `frame_num` para la imagen actual.

Una imagen que incluya una `memory_management_control_operation` de valor igual a 5 tendrá las restricciones que se aplican a `frame_num` descritas antes y, después de decodificar la imagen actual y de realizar las operaciones de control de gestión de memoria se supondrá que el valor `frame_num` de la imagen es 0 cuando se utilice posteriormente en el proceso de decodificación, salvo lo especificado en la subcláusula 7.4.1.2.4.

NOTA – Cuando la imagen codificada primaria no es una imagen IDR y no contiene el elemento sintáctico `memory_management_control_operation` de valor igual a 5, el valor de `frame_num` de una imagen codificada redundante correspondiente es el mismo que el valor de `frame_num` en la imagen codificada primaria. Otra posibilidad es que la imagen codificada redundante incluya un elemento sintáctico `memory_management_control_operation` de valor igual a 5 y la correspondiente imagen codificada primaria sea una imagen IDR.

**field\_pic\_flag** igual a 1 indica que el sector es un sector de un campo codificado. `field_pic_flag` igual a 0 indica que el sector es un sector de un cuadro codificado. Cuando no haya `field_pic_flag`, se supondrá que su valor es igual a 0.

La variable `MbaffFrameFlag` se calcula del siguiente modo:

$$\text{MbaffFrameFlag} = (\text{mb\_adaptive\_frame\_field\_flag} \ \&\& \ !\text{field\_pic\_flag}) \quad (7-11)$$

La variable altura de la imagen en unidades de macrobloque se calcula mediante la expresión:

$$\text{PicHeightInMbs} = \text{FrameHeightInMbs} / (1 + \text{field\_pic\_flag}) \quad (7-12)$$

La variable altura de la imagen correspondiente a la componente luma se calcula mediante la expresión:

$$\text{PicHeightInSamples}_L = \text{PicHeightInMbs} * 16 \quad (7-13)$$

La variable altura de la imagen correspondiente a la componente croma se calcula mediante la expresión:

$$\text{PicHeightInSamples}_C = \text{PicHeightInMbs} * 8 \quad (7-14)$$

La variable `PicSizeInMbs` de la imagen considerada se calcula de acuerdo con:

$$\text{PicSizeInMbs} = \text{PicWidthInMbs} * \text{PicHeightInMbs} \quad (7-15)$$

La variable `MaxPicNum` se calcula del siguiente modo:

- Si `field_pic_flag` es igual a 0, `MaxPicNum` es igual a `MaxFrameNum`.
- De lo contrario (`field_pic_flag` es igual a 1), `MaxPicNum` es  $2 * \text{MaxFrameNum}$ .

La variable `CurrPicNum` se calcula del modo siguiente:

- Si `field_pic_flag` es igual a 0, `CurrPicNum` es igual a `frame_num`.
- De lo contrario (`field_pic_flag` es igual a 1), `CurrPicNum` es igual a  $2 * \text{frame\_num} + 1$ .

**bottom\_field\_flag** igual a 1 indica que el sector forma parte de un campo inferior codificado. `bottom_field_flag` igual a 0 indica que la imagen es un campo superior codificado. Cuando el sector considerado no contiene este elemento sintáctico, se supondrá que su valor es 0.

**idr\_pic\_id** identifica una imagen IDR. Los valores de `idr_pic_id` en todos los sectores de una imagen IDR serán constantes. Cuando haya dos unidades de acceso consecutivas en el orden de decodificación que sean ambas unidades de acceso IDR, el valor de `idr_pic_id` en los sectores de la



primera de esas unidades de acceso IDR será diferente del `idr_pic_id` de la segunda de tales unidades de acceso IDR. El valor de `idr_pic_id` estará entre 0 y 65535, inclusive.

**`pic_order_cnt_lsb`** especifica el número de orden de la imagen módulo `MaxPicOrderCntLsb` correspondiente al campo superior de un cuadro codificado o a un campo codificado. El tamaño del elemento sintáctico `pic_order_cnt_lsb` es  $\log_2 \text{max\_pic\_order\_cnt\_lsb\_minus4} + 4$  bits. El valor de `pic_order_cnt_lsb` estará entre 0 y `MaxPicOrderCntLsb - 1`, inclusive.

**`delta_pic_order_cnt_bottom`** especifica la diferencia del número de orden de la imagen entre el campo inferior y el campo superior de un cuadro codificado, del modo siguiente:

- Si la imagen considerada incluye un `memory_management_control_operation` de valor igual a 5, el valor de `delta_pic_order_cnt_bottom` estará entre  $(1 - \text{MaxPicOrderCntLsb})$  y  $2^{31} - 1$ , inclusive.
- De lo contrario (la imagen considerada no incluye un `memory_management_control_operation` de valor igual a 5), el valor de `delta_pic_order_cnt_bottom` estará entre  $-2^{31}$  y  $2^{31} - 1$ , inclusive.

Cuando este elemento sintáctico no está en el tren de bits correspondiente al sector considerado, se supondrá que es igual a 0.

**`delta_pic_order_cnt[ 0 ]`** especifica la diferencia de número de orden de la imagen con respecto al número de orden de la imagen previsto correspondiente al campo superior de un cuadro codificado o a un campo codificado, según se especifica en la subcláusula 8.2.1. El valor de `delta_pic_order_cnt[ 0 ]` estará entre  $-2^{31}$  y  $2^{31} - 1$ , inclusive. Cuando este elemento sintáctico no aparezca en el tren de bits correspondiente al sector considerado, se supondrá que es igual a 0.

**`delta_pic_order_cnt[ 1 ]`** especifica la diferencia del número de orden de la imagen con respecto al número de orden de la imagen esperado correspondiente al campo inferior de un cuadro codificado, según se especifica en la subcláusula 8.2.1. El valor de `delta_pic_order_cnt[ 1 ]` estará entre  $-2^{31}$  y  $2^{31} - 1$ , inclusive. Cuando este elemento sintáctico no aparezca en el tren de bits correspondiente al sector considerado, se supondrá que es igual a 0.

**`redundant_pic_cnt`** será igual a 0 para los sectores y las particiones de datos de sector que pertenezcan a la imagen codificada primaria. El valor de `redundant_pic_cnt` será mayor que 0 para los sectores codificados o particiones de datos de sector codificados de una imagen codificada redundante. Cuando `redundant_pic_cnt` no aparezca en el tren de bits, se supondrá que su valor es 0. El valor de `redundant_pic_cnt` estará entre 0 y 127, inclusive.

NOTA – No debería haber una diferencia apreciable entre cualquier área de la imagen primaria decodificada y el área correspondiente que resultaría de aplicar el proceso de decodificación descrito en la cláusula 8 a cualquier imagen redundante de la misma unidad de acceso.

El valor de `pic_parameter_set_id` en un sector codificado o una partición de datos de sector codificado de una imagen codificada redundante será tal que el valor de `pic_order_present_flag` en el conjunto de parámetros imagen que se está utilizando en una imagen codificada redundante sea igual al valor de `pic_order_present_flag` en el conjunto de parámetros imagen que se está utilizando en la correspondiente imagen codificada primaria.

Cuando aparezcan en la imagen codificada primaria y en cualquier imagen codificada redundante, los siguientes elementos sintácticos tendrá el mismo valor: `field_pic_flag`, `bottom_field_flag` e `idr_pic_id`.

Cuando el valor de `nal_ref_idc` en una unidad NAL VCL de una unidad de acceso sea igual a 0, el valor de `nal_ref_idc` en las demás unidades NAL VCL de la misma unidad de acceso también será 0.

NOTA – La restricción anterior tiene además las siguientes repercusiones. Si el valor de `nal_ref_idc` en las unidades NAL VCL de la imagen codificada primaria es igual a 0, el valor de `nal_ref_idc` en

las unidades NAL VCL de las correspondientes imágenes codificadas redundantes también será 0; de lo contrario (el valor de `nal_ref_idc` en las unidades NAL VCL de la imagen codificada primaria es mayor que 0), el valor de `nal_ref_idc` en las unidades NAL VCL de las correspondientes imágenes codificadas redundantes también será mayor que 0.

El estado de marcado de imágenes de referencia y el valor de `frame_num` después de que se haya llamado al proceso de marcado de imágenes de referencia decodificadas descrito en la subcláusula 8.2.5 para la imagen codificada primaria y cualquier imagen codificada redundante de la misma unidad de acceso, serán idénticos independientemente de si se decodifica la imagen codificada primaria o cualquier imagen codificada redundante (en lugar de la imagen codificada primaria) de la unidad de acceso.

NOTA – La restricción anterior tiene además las siguientes repercusiones:

Si la imagen codificada primaria no es una imagen IDR, el contenido de la estructura sintáctica `dec_ref_pic_marking()` será idéntico en todas las cabeceras de sector de la imagen codificada primaria y en todas las imágenes codificadas redundantes correspondientes a la imagen codificada primaria.

De lo contrario (la imagen codificada primaria es una imagen IDR) se aplica a lo siguiente.

Si la imagen codificada redundante correspondiente a la imagen codificada primaria es una imagen IDR, el contenido de la estructura sintáctica `dec_ref_pic_marking()` debe ser idéntico en todas las cabeceras de sector de la imagen codificada primaria y de la imagen codificada redundante correspondiente a la imagen codificada primaria.

De lo contrario (la imagen redundante correspondiente a la imagen codificada primaria no es una imagen IDR) todas las cabeceras de sector de la imagen redundante deben contener la estructura sintáctica `dec_ref_pic_marking syntax()` incluido el elemento sintáctico `memory_management_control_operation` de valor igual a 5, y se aplica lo siguiente.

Si el valor de `long_term_reference_flag` en la imagen codificada primaria es igual a 0, la estructura sintáctica `dec_ref_pic_marking` de la imagen codificada redundante no debe incluir el elemento sintáctico `memory_management_control_operation` con valor igual a 6.

De lo contrario (el valor de `long_term_reference_flag` en la imagen codificada primaria es igual a 1), la estructura sintáctica `dec_ref_pic_marking` de la imagen codificada redundante debe incluir elementos sintácticos `memory_management_control_operation` con valor igual a 5, 4 y 6 en orden de decodificación, y el valor de `max_long_term_frame_idx_plus1` debe ser igual a 1, y el valor de `long_term_frame_idx` debe ser igual a 0.

Los valores de `TopFieldOrderCnt` y `BottomFieldOrderCnt` (si los hubiere) que resultan de aplicar el proceso de decodificación a cualquier imagen codificada redundante o a la imagen codificada primaria de la misma unidad de acceso serán idénticos independientemente de si se decodifica la imagen codificada primaria o cualquier imagen codificada redundante (en lugar de la imagen codificada primaria) de la unidad de acceso.

No hay un proceso de decodificación obligatorio aplicable a sectores codificados o partición de datos de sectores codificados de una imagen codificada redundante. Cuando `redundant_pic_cnt` en la cabecera de sector de un sector codificado es mayor que 0, el decodificador puede descartar el sector codificado. Ahora bien, el sector codificado o la partición de datos de sector codificado de cualquier imagen codificada redundante debe cumplir las mismas restricciones que el sector codificado o la partición de datos de sector codificado de una imagen primaria.

NOTA – Cuando algunas muestras de la imagen primaria decodificada no se puedan decodificar correctamente a causa de errores o pérdidas en la transmisión de la secuencia y, en cambio, si se pueda decodificar correctamente el sector redundante codificado, el decodificador debería reemplazar las muestras de la imagen primaria decodificada con las correspondientes muestras del sector redundante decodificado. Cuando haya más de un sector redundante que abarque la región pertinente de la imagen primaria, se debería utilizar el sector redundante que tenga el valor más pequeño de `redundant_pic_cnt`.

Los sectores y las particiones de datos de sector redundantes que tengan el mismo valor de `redundant_pic_cnt` pertenecen a la misma imagen redundante. Los sectores decodificados dentro de la misma imagen redundante no necesitan abarcar toda el área de la imagen y no deben solaparse.

**direct\_spatial\_mv\_pred\_flag** especifica el método utilizado en el proceso de decodificación para obtener los vectores de movimiento y los índices de referencia para la predicción inter, del modo siguiente:

- Si `direct_spatial_mv_pred_flag` es igual a 1, se utilizará el modo de predicción directa espacial descrito en la subcláusula 8.4.1.2.2 para el proceso de obtención de vectores de movimiento luma correspondientes a `B_Skip`, `B_Direct_16x16` y `B_Direct_8x8` de la subcláusula 8.4.1.2.
- De lo contrario (`direct_spatial_mv_pred_flag` es igual a 0), se utilizará el modo de predicción directa temporal descrito en la subcláusula 8.4.1.2.3 para el proceso de obtención de vectores de movimiento luma correspondientes a `B_Skip`, `B_Direct_16x16` y `B_Direct_8x8` de la subcláusula 8.4.1.2.

**num\_ref\_idx\_active\_override\_flag** igual a 0 indica que los valores de los elementos sintácticos `num_ref_idx_l0_active_minus1` y `num_ref_idx_l1_active_minus1` especificados en el conjunto de parámetros imagen referenciados están activos. `num_ref_idx_active_override_flag` igual a 1 indica que `num_ref_idx_l0_active_minus1` y `num_ref_idx_l1_active_minus1` especificados en el conjunto de parámetros imagen referenciados se han sustituido en el sector considerado (y únicamente en el sector considerado) por los siguientes valores en la cabecera de sector.

Cuando el sector considerado es un sector P, SP o B y `field_pic_flag` es igual a 0 y el valor de `num_ref_idx_l0_active_minus1` en el conjunto de parámetros de imagen es mayor que 15, `num_ref_idx_active_override_flag` será igual a 1.

Cuando el sector considerado es un sector B y `field_pic_flag` es igual a 0 y el valor de `num_ref_idx_l1_active_minus1` en el conjunto de parámetros de imagen es mayor que 15, `num_ref_idx_active_override_flag` será igual a 1.

**num\_ref\_idx\_l0\_active\_minus1** especifica el índice de referencia máximo de la lista 0 de imagen de referencia que se utilizará para decodificar el sector.

Los valores posibles de `num_ref_idx_l0_active_minus1` se indican a continuación.

- Si `field_pic_flag` es igual a 0, `num_ref_idx_l0_active_minus1` estará entre 0 y 15, inclusive. Cuando `MbaffFrameFlag` es igual a 1, `num_ref_idx_l0_active_minus1` es el valor máximo del índice para la decodificación de macrobloques de cuadro y  $2 * \text{num\_ref\_idx\_l0\_active\_minus1} + 1$  es el valor máximo del índice para la decodificación de macrobloques campo.
- De lo contrario (`field_pic_flag` es igual a 1), `num_ref_idx_l0_active_minus1` estará entre 0 y 31, inclusive.

**num\_ref\_idx\_l1\_active\_minus1** tiene la misma semántica que `num_ref_idx_l0_active_minus1`, sustituyendo l0 y lista 0 por l1 y lista 1, respectivamente.

**cabac\_init\_idc** especifica el índice para determinar la tabla de inicialización utilizada en el proceso de inicialización de variables de contexto. El valor de `cabac_init_idc` estará entre 0 y 2, inclusive.

**slice\_qp\_delta** especifica el valor inicial de  $QP_Y$  que se utilizará para todos los macrobloques del sector hasta que el valor de `mb_qp_delta` en la capa de macrobloque lo modifique. El valor inicial del parámetro de cuantización  $QP_Y$  para el sector se calcula mediante la expresión:

$$\text{SliceQP}_Y = 26 + \text{pic\_init\_qp\_minus26} + \text{slice\_qp\_delta} \quad (7-16)$$

El valor de `slice_qp_delta` se limitará de modo que el valor de  $QP_Y$  esté entre 0 y 51, inclusive.

**sp\_for\_switch\_flag** especifica el proceso de decodificación que se utilizará para decodificar macrobloques P en un sector SP, del modo siguiente:

- Si **sp\_for\_switch\_flag** es igual a 0, los macrobloques P en el sector SP se decodificarán utilizando el proceso de decodificación SP aplicable a imágenes no intercambiados, descrito en la subcláusula 8.6.1.
- De lo contrario (**sp\_for\_switch\_flag** es igual a 1), los macrobloques P en el sector SP se decodificarán utilizando el proceso de decodificación SP y SI aplicable a imágenes intercambiadas, descrito en la subcláusula 8.6.2.

**slice\_qs\_delta** especifica el valor de  $QS_Y$  para todos los macrobloques en los sectores SP y SI. El valor del parámetros de cuantificación  $QS_Y$  para el sector se calcula del modo siguiente:

$$QS_Y = 26 + pic\_init\_qs\_minus26 + slice\_qs\_delta \quad (7-17)$$

El valor de **slice\_qs\_delta** se limitará para que el valor de  $QS_Y$  esté entre 0 y 51, inclusive. Este valor de  $QS_Y$  se utiliza para la decodificación de todos los macrobloques en sectores SI con **mb\_type** igual a SI y todos los macrobloques en sectores SP con modo predicción Inter.

**disable\_deblocking\_filter\_idc** especifica si el filtrado de bloques se deshabilitará en algunos bordes del bloque del sector y especifica en qué bordes se deshabilita el filtrado. Cuando no haya **disable\_deblocking\_filter\_idc** en la cabecera del sector, se considerará que el valor de **disable\_deblocking\_filter\_idc** es 0.

El valor de **disable\_deblocking\_filter\_idc** estará entre 0 y 2, inclusive.

**slice\_alpha\_c0\_offset\_div2** especifica la traslación utilizada al acceder a los cuadros de filtro de bloques  $\alpha$  y  $t_{c0}$  aplicables al filtrado controlado por los macrobloques dentro del sector. La traslación que se aplica cuando se acceda a estos cuadros se calcula a partir de este valor mediante la expresión:

$$FilterOffsetA = slice\_alpha\_c0\_offset\_div2 \ll 1 \quad (7-18)$$

El valor de **slice\_alpha\_c0\_offset\_div2** estará entre  $-6$  y  $+6$ , inclusive. Cuando no haya **slice\_alpha\_c0\_offset\_div2** en la cabecera del sector, el valor de **slice\_alpha\_c0\_offset\_div2** se supondrá igual a 0.

**slice\_beta\_offset\_div2** especifica la traslación utilizada al acceder al cuadro de filtro de bloque  $\beta$  para el filtrado controlado por los macrobloques dentro del sector. La traslación que se aplica cuando se accede al cuadro  $\beta$  de filtrado de desbloqueo se calcula a partir de este valor mediante la expresión:

$$FilterOffsetB = slice\_beta\_offset\_div2 \ll 1 \quad (7-19)$$

El valor de **slice\_beta\_offset\_div2** estará entre  $-6$  y  $+6$ , inclusive. Cuando no haya **slice\_beta\_offset\_div2** en la cabecera del sector, el valor de **slice\_beta\_offset\_div2** se supondrá igual a 0.

**slice\_group\_change\_cycle** se utiliza para calcular el número de unidades de mapeado de grupos de sector en el grupo 0 de sector cuando **slice\_group\_map\_type** es igual a 3, 4 ó 5, según la siguiente expresión:

$$MapUnitsInSliceGroup0 = \text{Min}( slice\_group\_change\_cycle * SliceGroupChangeRate, PicSizeInMapUnits ) \quad (7-20)$$

El valor de **slice\_group\_change\_cycle** está representado en el flujo de bits mediante el siguiente número de bits:

$$\text{Ceil}( \text{Log2}( PicSizeInMapUnits \div SliceGroupChangeRate + 1 ) ) \quad (7-21)$$

El valor de **slice\_group\_change\_cycle** estará entre 0 y  $\text{Ceil}( PicSizeInMapUnits \div SliceGroupChangeRate )$ , inclusive.

### 7.4.3.1 Semántica de reordenación de listas de imágenes de referencia

Los elementos sintácticos `reordering_of_pic_nums_idc`, `abs_diff_pic_num_minus1` y `long_term_pic_num` especifican el cambio de las listas de imágenes de referencia iniciales a las listas de imágenes de referencia que se utilizarán para decodificar el sector.

`ref_pic_list_reordering_flag_l0` igual a 1 indica que el elemento sintáctico `reordering_of_pic_nums_idc` está presente y que indica la lista 0 de imágenes de referencia. `ref_pic_list_reordering_flag_l0` igual a 0 indica que este elemento sintáctico no está presente.

Cuando `ref_pic_list_reordering_flag_l0` es igual a 1, el número de veces que `reordering_of_pic_nums_idc` es distinto de 3 después de `ref_pic_list_reordering_flag_l0` no será mayor que `num_ref_idx_l0_active_minus1 + 1`.

Cuando `RefPicList0[num_ref_idx_l0_active_minus1]` en la lista de imágenes de referencia inicial, generada como se especifica en la subcláusula 8.2.4.2, es igual a "imagen que no es de referencia", `ref_pic_list_reordering_flag_l0` será igual a 1 y `reordering_of_pic_nums_idc` será distinto de 3 hasta que `RefPicList0[num_ref_idx_l0_active_minus1]` en la lista reordenada, generada según se especifica en la subcláusula 8.2.4.3, sea distinto de "imagen que no es de referencia".

`ref_pic_list_reordering_flag_l1` igual a 1 indica que el elemento sintáctico `reordering_of_pic_nums_idc` está presente y que especifica la lista 1 de imágenes de referencia. `ref_pic_list_reordering_flag_l1` igual a 0 indica que este elemento sintáctico no está presente.

Cuando `ref_pic_list_reordering_flag_l1` es igual a 1, el número de veces que `reordering_of_pic_nums_idc` es distinto de 3 después de `ref_pic_list_reordering_flag_l1` no sobrepasará `num_ref_idx_l1_active_minus1 + 1`.

Cuando se decodifica un sector B y `RefPicList1[num_ref_idx_l1_active_minus1]` en la lista de imágenes de referencia inicial generada según se describe en la subcláusula 8.2.4.2 es igual a "imagen que no es de referencia", `ref_pic_list_reordering_flag_l1` será igual a 1 y `reordering_of_pic_nums_idc` será distinto de 3 hasta que `RefPicList1[num_ref_idx_l1_active_minus1]` en la lista reordenada generada según se describe en la subcláusula 8.2.4.3 sea distinta de "imagen que no es de referencia".

`reordering_of_pic_nums_idc` junto con `abs_diff_pic_num_minus1` o `long_term_pic_num` indica cuál de las imágenes de referencia se ha vuelto a mapear. Los valores de `reordering_of_pic_nums_idc` se especifican en el cuadro 7-4. El valor del primer `reordering_of_pic_nums_idc` que esté inmediatamente después de `ref_pic_list_reordering_flag_l0` o `ref_pic_list_reordering_flag_l1` será distinto de 3.

**Cuadro 7-4 – Operaciones `reordering_of_pic_nums_idc` para reordenar las listas de imágenes de referencia**

<code>reordering_of_pic_nums_idc</code>	Reordenación especificada
0	<code>abs_diff_pic_num_minus1</code> está presente y corresponde al valor que se ha de restar del valor de predicción del número de imagen
1	<code>abs_diff_pic_num_minus1</code> está presente y corresponde al valor que se ha de sumar al valor de predicción del número de imagen
2	<code>long_term_pic_num</code> está presente y especifica el número de imágenes de largo alcance para una imagen de referencia
3	Fin del bucle de reordenación de la lista de imágenes de referencia inicial

**abs\_diff\_pic\_num\_minus1** más 1 indica la diferencia en valor absoluto entre el número de imagen de la imagen a la que apunta el índice considerado en la lista y el valor de predicción del número de imagen. El valor de **abs\_diff\_pic\_num\_minus1** estará entre 0 y  $\text{MaxPicNum} - 1$ . Los valores permitidos de **abs\_diff\_pic\_num\_minus1** se restringen aún más en la subcláusula 8.2.4.3.1.

**long\_term\_pic\_num** especifica el número de imagen de largo alcance de la imagen a la que apunta el índice considerado en la lista. Al decodificar un cuadro codificado, **long\_term\_pic\_num** será igual al **LongTermPicNum** asignado a uno de los cuadros de referencia o a un par de campos de referencia complementarios, marcados "utilizado para referencia de largo alcance". Al decodificar un campo codificado, **long\_term\_pic\_num** será igual al **LongTermPicNum** asignado a uno de los campos de referencia marcado como "utilizado para referencia de largo alcance".

#### 7.4.3.2 Semántica del cuadro de ponderaciones de predicción

**luma\_log2\_weight\_denom** es el logaritmo en base 2 del denominador de todos los factores de ponderación luma. El valor de **luma\_log2\_weight\_denom** estará entre 0 y 7, inclusive.

**chroma\_log2\_weight\_denom** es el logaritmo en base 2 del denominador de todos los factores de ponderación croma. El valor de **chroma\_log2\_weight\_denom** estará entre 0 y 7, inclusive.

**luma\_weight\_10\_flag** igual a 1 indica que hay factores de ponderación correspondientes a la componente luma de la predicción de lista 0. **luma\_weight\_10\_flag** igual a 0 indica que no hay estos factores de ponderación.

**luma\_weight\_10[i]** es el factor de ponderación aplicado al valor de predicción luma correspondiente a la predicción de la lista 0 utilizando **RefPicList0[i]**. Si **luma\_weight\_10\_flag** es igual a 1, el valor de **luma\_weight\_10[i]** estará entre  $-128$  y  $127$ , inclusive. Cuando **luma\_weight\_10\_flag** sea igual a 0, se supondrá que el valor de **luma\_weight\_10[i]** es  $2^{\text{luma\_log2\_weight\_denom}}$  para **RefPicList0[i]**.

**luma\_offset\_10[i]** es la traslación adicional que se aplica al valor de predicción luma para la predicción de lista 0 utilizando **RefPicList0[i]**. El valor de **luma\_offset\_10[i]** estará entre  $-128$  y  $127$ , inclusive. Cuando **luma\_weight\_10\_flag** sea igual a 0, se supondrá que el valor de **luma\_offset\_10[i]** es 0 para **RefPicList0[i]**.

**chroma\_weight\_10\_flag** igual a 1 indica que los factores de ponderación aplicables a los valores de predicción croma de la predicción de lista 0 están presentes. **chroma\_weight\_10\_flag** igual a 0 indica que estos factores de ponderación no están presentes.

**chroma\_weight\_10[i][j]** es el factor de ponderación aplicable a los valores de predicción croma de la predicción de la lista 0 que utiliza **RefPicList0[i]**, con  $j$  igual a 0 para Cb y  $j$  igual a 1 para Cr. Si **chroma\_weight\_10\_flag** es igual a 1, el valor de **chroma\_weight\_10[i][j]** estará entre  $-128$  y  $127$ , inclusive. Cuando **chroma\_weight\_10\_flag** sea igual a 0, se supondrá que el valor de **chroma\_weight\_10[i][j]** es  $2^{\text{chroma\_log2\_weight\_denom}}$  para **RefPicList0[i]**.

**chroma\_offset\_10[i][j]** es la traslación adicional que se aplica a los valores de predicción croma para la predicción de lista 0 utilizando **RefPicList0[i]**, con  $j$  igual a 0 para Cb y  $j$  igual a 1 para Cr. El valor de **chroma\_offset\_10[i][j]** estará entre  $-128$  y  $127$ , inclusive. Cuando **chroma\_weight\_10\_flag** sea igual a 0, se supondrá que el valor de **chroma\_offset\_10[i][j]** es 0 para **RefPicList0[i]**.

**luma\_weight\_11\_flag**, **luma\_weight\_11**, **luma\_offset\_11**, **chroma\_weight\_11\_flag**, **chroma\_weight\_11**, **chroma\_offset\_11** tienen las mismas semánticas que **luma\_weight\_10\_flag**, **luma\_weight\_10**, **luma\_offset\_10**, **chroma\_weight\_10\_flag**, **chroma\_weight\_10**, **chroma\_offset\_10**, sustituyendo 10, list 0 y List0 por 11, list 1 y List1, respectivamente.

### 7.4.3.3 Semántica de marcado de imágenes de referencia decodificadas

Los elementos sintácticos `no_output_of_prior_pics_flag`, `long_term_reference_flag`, `adaptive_ref_pic_marking_mode_flag`, `memory_management_control_operation`, `difference_of_pic_nums_minus1`, `long_term_frame_idx`, `long_term_pic_num`, y `max_long_term_frame_idx_plus1` especifican el marcado de imágenes de referencia.

El marcado de una imagen de referencia puede ser "no utilizada como referencia", "utilizada para referencia de corto alcance", o "utilizada para referencia de largo alcance", y está limitado a uno de estos tres. Cuando se dice que una imagen de referencia está marcada como "utilizada como referencia" significa que la imagen está marcada como "utilizada para referencia de corto alcance" o "utilizada para referencia de largo alcance", pero no ambas.

El elemento sintáctico `adaptive_ref_pic_marking_mode_flag` y el contenido de la estructura sintáctica marcado de imágenes de referencia decodificadas será idéntico para todos los sectores codificados de una imagen codificada.

La categoría sintáctica de la estructura sintáctica marcado de imágenes de referencia decodificadas se deducirá del modo siguiente:

- Si la estructura sintáctica marcado de imágenes de referencia decodificadas está en una cabecera de sector, la categoría sintáctica de la estructura sintáctica marcado de imágenes de referencia decodificadas se supondrá que es 2.
- De lo contrario (la estructura sintáctica de marcado imágenes de referencia decodificadas está en un mensaje SEI de repetición de marcado de imágenes de referencia decodificadas, como se describe en el anexo D) se supondrá que la categoría sintáctica de la estructura sintáctica marcado de imágenes de referencia decodificadas es igual a 5.

**`no_output_of_prior_pics_flag`** especifica cómo se tratan las imágenes decodificadas previamente en la memoria intermedia de imágenes decodificadas después de decodificar una imagen IDR. Véase anexo C. Cuando la imagen IDR es la primera imagen IDR del tren de bits, el valor de `no_output_of_prior_pics_flag` no afecta al proceso de decodificación. Cuando la imagen IDR no es la primera imagen IDR del tren de bits y el valor de `PicWidthInMbs`, `FrameHeightInMbs` o `max_dec_frame_buffering` calculado a partir del conjunto de parámetros secuencia activo es diferente del valor de `PicWidthInMbs`, `FrameHeightInMbs` o `max_dec_frame_buffering` calculado a partir del conjunto de parámetros secuencia activo para la secuencia precedente, el decodificador puede suponer que el valor de `no_output_of_prior_pics_flag` es igual a 1, independientemente del valor real de `no_output_of_prior_pics_flag`.

**`long_term_reference_flag`** igual a 0 indica que la variable `MaxLongTermFrameIdx` es igual a "no hay índices de cuadro de largo alcance" y que la imagen IDR está marcada como "utilizada para referencia de corto alcance". `long_term_reference_flag` igual a 1 indica que el valor de la variable `MaxLongTermFrameIdx` es 0 y que la imagen IDR considerada está marcada como "utilizada para referencia de largo alcance" y que `LongTermFrameIdx` es igual a 0. Si `num_ref_frames` es igual a 0, `long_term_reference_flag` será igual a 0.

**`adaptive_ref_pic_marking_mode_flag`** selecciona el modo de marcado de imágenes de referencia de la imagen que se está decodificando según se especifica en el cuadro 7-5. `adaptive_ref_pic_marking_mode_flag` será igual a 1 cuando el número de cuadros, pares de campos complementarios y campos desapareados que están marcados como "utilizados para referencia de largo alcance" es igual a  $\text{Max}(\text{num\_ref\_frames}, 1)$ .

**Cuadro 7-5 – Interpretación de adaptive\_ref\_pic\_marking\_mode\_flag**

adaptive_ref_pic_marking_mode_flag	Modo de marcado de imágenes de referencia especificado
0	Modo de marcado de imágenes de referencia de ventana deslizante: modo de marcado que consiste en un mecanismo primero en entrar, primero en salir de imágenes de referencia de corto alcance.
1	Modo de marcado de imágenes de referencia adaptativo: modo de marcado de imágenes de referencia que utiliza elementos sintácticos para marcar imágenes de referencia como "no utilizado como referencia" y para asignar índices de cuadro de largo alcance.

**memory\_management\_control\_operation** especifica el control que se aplica a gestión del marcado de imágenes de referencia. El elemento sintáctico **memory\_management\_control\_operation** viene seguido por los datos necesarios para la operación especificada por el valor de **memory\_management\_control\_operation**. Los valores y las operaciones de control relacionadas con **memory\_management\_control\_operation** se especifican en el cuadro 7-6. El proceso de decodificación procesa los elementos sintácticos **memory\_management\_control\_operation** en el mismo orden en el que aparecen en la cabecera de sector, y se aplican las restricciones semánticas especificadas para cada **memory\_management\_control\_operation** en la posición específica en el orden en que se procesa esa **memory\_management\_control\_operation**.

**memory\_management\_control\_operation** será distinto de 1 en la cabecera de sector, a no ser que la imagen de corto alcance especificada esté marcada como "utilizada como referencia" cuando el proceso de decodificación procese la **memory\_management\_control\_operation**, y la imagen de corto alcance en cuestión no tenga asignado un índice de cuadro de largo alcance y no esté asignada a un índice de cuadro de largo alcance en la misma estructura sintáctica marcado de imágenes de referencia decodificadas.

**memory\_management\_control\_operation** será distinto de 2 en la cabecera del sector, a no ser que el número de imagen de largo alcance especificado se refiera a un cuadro o campo que esté marcado como "utilizado como referencia" cuando el proceso de decodificación procese la **memory\_management\_control\_operation**.

**memory\_management\_control\_operation** será distinto de 3 en la cabecera de sector, a no ser que la imagen de referencia de corto alcance especificada esté marcada actualmente como "utilizada como referencia" cuando el proceso de decodificación procese la **memory\_management\_control\_operation** y a la imagen de referencia de corto alcance no se le haya asignado previamente un índice de cuadro de largo alcance y no esté asignada a ningún otro índice cuadro de largo alcance dentro de la misma estructura sintáctica marcado de imágenes de referencia decodificadas.

**memory\_management\_control\_operation** será diferente de 3 o de 6 cuando el valor de la variable **MaxLongTermFrameIdx** sea igual a "no hay índices de cuadro de largo alcance" cuando el proceso de decodificación procese la **memory\_management\_control\_operation**.

En una cabecera de sector no habrá más de un **memory\_management\_control\_operation** igual a 4.

**memory\_management\_control\_operation** será distinto de 5 en la cabecera de sector, a no ser que no haya ningún **memory\_management\_control\_operation** cuyo valor esté comprendido entre 1 y 3 en la misma estructura sintáctica marcado de imágenes de referencia decodificadas.

En una cabecera de sector no habrá más de una **memory\_management\_control\_operation** igual a 6.



Cuando una `memory_management_control_operation` sea igual a 6, las `memory_management_control_operation` iguales a 2, 3 ó 4 que la sigan dentro de la misma cabecera de sector no especificarán que la imagen actual está marcada como "no utilizada como referencia".

Una `memory_management_control_operation` igual a 6 no puede anteceder a una `memory_management_control_operation` igual a 5 dentro de la misma cabecera de sector.

NOTA – Estas restricciones prohíben cualquier combinación de varios elementos sintácticos `memory_management_control_operation` que especificarían que la imagen actual está marcada como "no utilizada como referencia". Sin embargo se permiten otras combinaciones de elementos sintácticos `memory_management_control_operation` que podrían tener una incidencia, en más de una ocasión dentro de una misma cabecera de sector, en la forma en que se marcan otras imágenes de referencia. En particular, dentro de una misma cabecera de sector, una `memory_management_control_operation` igual a 2, 3 ó 4 que especifica que la misma imagen de referencia sea en lo subsiguiente marcada como "no utilizada como referencia" puede seguir a una `memory_management_control_operation` igual a 3 que especifica que un índice de cuadro de largo alcance se asigne a una imagen de referencia de corto alcance en particular.

En la cabecera de sector no habrá más de un `memory_management_control_operation` que especifique la misma acción.

**Cuadro 7-6 – Valores de operación de control de gestión de memoria (`memory_management_control_operation`)**

<code>memory_management_control_operation</code>	Operación de control de gestión de memoria
0	Fin del bucle <code>memory_management_control_operation</code>
1	Marca la imagen de corto alcance como "no utilizada como referencia"
2	Marca el cuadro o campo que tiene un número de imagen de largo alcance como "no utilizado como referencia"
3	Asigna un índice de cuadro de largo alcance a imágenes de corto alcance
4	Especifica el valor máximo del índice de cuadro de largo alcance.
5	Marca todas las imágenes de referencia como "no utilizada como referencia" y fija el valor de la variable <code>MaxLongTermFrameIdx</code> a "no hay índices de cuadro de largo alcance"
6	Asigna el índice de cuadro de largo alcance a la imagen decodificada actual

Al decodificar un campo y una instrucción `memory_management_control_operation` de valor 3 asigna un índice de cuadro de largo alcance a un campo que forma parte de un cuadro de referencia de corto alcance o un par de campos de referencia complementarios de corto alcance, deberá haber en la misma estructura de sintáctica marcado de imágenes de referencia decodificadas otra instrucción `memory_management_control_operation` que asigne el mismo índice de cuadro de largo alcance al otro campo del mismo cuadro o par de campos de referencia complementario.

Cuando el primer campo (en el orden de decodificación) de un par de campos de referencia complementario incluya un `long_term_reference_flag` igual a 1 o una instrucción `memory_management_control_operation` igual a 6, la estructura sintáctica marcado de imágenes de referencia decodificadas del otro campo del par de campos de referencia complementarios

contendrá una instrucción `memory_management_control_operation` igual a 6 que asigne el mismo índice de cuadro de largo alcance al otro campo.

**difference\_of\_pic\_nums\_minus1** se utiliza (siendo `memory_management_control_operation` igual a 3 ó 1) para asignar un índice de cuadro de largo alcance a una imagen de referencia de corto alcance o para marcar una imagen de referencia de corto alcance como "no utilizada como referencia". Cuando el proceso de decodificación procese la `memory_management_control_operation` asociada, el número de imagen resultante obtenido a partir de `difference_of_pic_nums_minus1` será un número de imagen asignado a una de las imágenes de referencia marcadas como "utilizada como referencia" y que no se haya asignado previamente a un índice de cuadro de largo alcance.

El número de imagen de largo alcance resultante debe cumplir con lo siguiente:

- Si `field_pic_flag` es igual a 0, el número de imagen resultante será uno del conjunto de números de imagen asignado a cuadros de referencia o pares de campos de referencia complementarios.

NOTA – Si `field_pic_flag` es igual a 0, el número de imagen de largo alcance resultante debe ser un número de imagen de largo alcance asignado a un par de campos de referencia complementarios en el que ambos campos están marcados como "utilizado como referencia" o un cuadro en el que ambos campos están marcados como "utilizado como referencia". En particular, una `memory_management_control_operation` igual a 1 no puede afectar la marcación de un campo que no ha sido apareado o un cuadro en el que solamente un campo ha sido marcado como "utilizado como referencia", cuando `field_pic_flag` es igual a 0.

- De lo contrario (`field_pic_flag` es igual a 1), el número de imagen resultante será uno del conjunto de números de imágenes asignados a campos de referencia.

**long\_term\_pic\_num** se utiliza para marcar las imágenes de referencia de largo alcance como "no utilizadas como referencia" (siendo `memory_management_control_operation` igual a 2). `long_term_pic_num` será igual al número de imágenes de largo alcance asignado a una de las imágenes de referencia marcada como "utilizada como referencia de largo alcance", cuando el proceso de decodificación procesa la `memory_management_control_operation` asociada.

El número de imagen de largo alcance resultante debe cumplir con lo siguiente:

- Si `field_pic_flag` es igual a 0, el número de imagen de largo alcance resultante será uno del conjunto de números de imagen de largo alcance asignados a cuadros de referencia o a pares de campos de referencia complementarios.

NOTA – Si `field_pic_flag` es igual a 0, el número de imagen de largo alcance resultante debe ser un número de imagen de largo alcance resultante asignado a un par de campos de referencia complementarios en el que ambos campos están marcados como "utilizado como referencia" o un cuadro en el que ambos campos están marcados como "utilizado como referencia". En particular, una `memory_management_control_operation` igual a 2 no puede afectar la marcación de un campo que no ha sido apareado o un cuadro en el que solamente un campo ha sido marcado como "utilizado como referencia", cuando `field_pic_flag` es igual a 0.

- De lo contrario (`field_pic_flag` es igual a 1), el número de imagen de largo alcance resultante será uno del conjunto de números de imagen de largo alcance asignados a campos de referencia.

**long\_term\_frame\_idx** se utiliza (siendo `memory_management_control_operation` igual a 3 ó 6) para asignar un índice de cuadro de largo alcance a una imagen. Cuando el proceso de decodificación procese la `memory_management_control_operation` asociada, el valor de `long_term_frame_idx` estará entre 0 y `MaxLongTermFrameIdx`, inclusive.

**max\_long\_term\_frame\_idx\_plus1** menos 1 especifica el valor máximo del índice de cuadro de largo alcance permitido para las imágenes de referencia de largo alcance (hasta que se reciba otro

valor de `max_long_term_frame_idx_plus1`). El valor de `max_long_term_frame_idx_plus1` estará entre 0 y `num_ref_frames`, inclusive.

#### 7.4.4 Semántica de datos de sector

**`cabac_alignment_one_bit`** es un bit de valor 1.

**`mb_skip_run`** especifica el número de macrobloques obviados consecutivos para los cuales, al decodificar un sector P o SP, se supondrá que `mb_type` es `P_Skip` y el tipo de macrobloque se denomina generalmente como macrobloque de tipo P, y, en cambio, al decodificar un sector B, se supondrá que el `mb_type` es `B_Skip` y el tipo de macrobloque se denomina generalmente como macrobloque de tipo B. El valor de `mb_skip_run` estará entre 0 y `PicSizeInMbs - CurrMbAddr`, inclusive.

**`mb_skip_flag`** igual a 1 indica que para el macrobloque considerado, al decodificar un sector P o un sector SP, se supondrá que `mb_type` es `P_Skip` y que el tipo de macrobloque se denomina genéricamente como macrobloque de tipo P, y al decodificar un sector B, se supondrá que `mb_type` es `B_Skip` y el tipo de macrobloque se denomina genéricamente como macrobloque de tipo B. `mb_skip_flag` igual a 0 indica que no se salta el macrobloque considerado.

**`mb_field_decoding_flag`** igual a 0 indica que el par de macrobloques considerado es un par de macrobloques cuadro. `mb_field_decoding_flag` igual a 1 indica que el par de macrobloques es un par de macrobloques campo. Los dos macrobloques de un par de macrobloques cuadro aparecen denominados en el texto como macrobloques cuadro, y los dos macrobloques de un par de macrobloques campo aparecen denominados en el texto como macrobloques campo.

Cuando no haya `mb_field_decoding_flag` para algún macrobloque de un par de macrobloques, el valor de `mb_field_decoding_flag` se calcula del modo siguiente:

- Si hay un par de macrobloques adyacentes justo a la izquierda del par de macrobloques considerado en el mismo sector, se tomará como valor de `mb_field_decoding_flag` el valor de `mb_field_decoding_flag` del par de macrobloques adyacentes pegado a la izquierda del par de macrobloques considerado.
- De lo contrario, si no hay ningún par de macrobloques adyacentes justo a la izquierda del par de macrobloques considerado en el mismo sector y hay un par de macrobloques adyacente justo arriba del par de macrobloques considerado en el mismo sector, se tomará como valor de `mb_field_decoding_flag` el valor del `mb_field_decoding_flag` del par de macrobloques adyacentes justo arriba del par de macrobloques considerado.
- De lo contrario (si no hay ningún par de macrobloques adyacente que esté pegado a la izquierda o justo arriba del par de macrobloques considerado en el mismo sector), se supondrá que el valor de `mb_field_decoding_flag` es igual a 0.

**`end_of_slice_flag`** igual a 0 indica que hay otro macrobloque que viene a continuación en el sector. `end_of_slice_flag` igual a 1 indica el final del sector y que no hay más macrobloques.

La función `NextMbAddress()` que se utiliza en el cuadro de sintaxis de datos de sector se describe en la subcláusula 8.2.2.

#### 7.4.5 Semántica de la capa macrobloques

**`mb_type`** especifica el tipo de macrobloque. La semántica de `mb_type` depende del tipo de sector.

En los cuadros y la semántica que figuran a continuación se especifican los diversos tipos de macrobloques para sectores I, SI, P, SP, y B. Cada cuadro presenta el valor de `mb_type`, en nombre de `mb_type`, el número de particiones macrobloque utilizadas (dado por la función `NumMbPart(mb_type)`), el modo de predicción del macrobloque (si no está subdividido en particiones) o de la primera partición (dado por la función `MbPartPredMode(mb_type, 0)`) y el modo de predicción de la segunda partición (dado por la función `MbPartPredMode(mb_type, 1)`).

Cuando un valor no es aplicable se indica mediante "na". El valor de mb\_type puede aparecer denominado en el texto como tipo de macrobloque y el valor X de MbPartPredMode( ) puede aparecer denominado en el texto como "modo de predicción (de partición) del macrobloque X" o como "macrobloques de predicción X".

En el cuadro 7-7 se indican los posibles tipos de macrobloques genéricos para cada slice\_type.

NOTA – Hay ciertos tipos de macrobloque con modo de predicción Pred\_L0 que se clasifican como macrobloques de tipo B.

**Cuadro 7-7 – Tipos de macrobloques genéricos posibles en función de slice\_type**

<b>slice_type</b>	<b>Tipos de macrobloques genéricos posibles</b>
I (sector)	I (véase cuadro 7-8) (tipos de macrobloque)
P (sector)	P (véase cuadro 7-10) e I (véase cuadro 7-8) (tipos de macrobloque)
B (sector)	B (véase cuadro 7-11) e I (véase cuadro 7-8) (tipos de macrobloque)
SI (sector)	SI (véase cuadro 7-9) e I (véase cuadro 7-8) (tipos de macrobloque)
SP (sector)	P (véase cuadro 7-10) e I (véase cuadro 7-8) (tipos de macrobloque)

En el cuadro 7-8 se especifican los tipos de macrobloque que pueden aparecer denominados genéricamente como macrobloques de tipo I.

Los tipos de macrobloques correspondientes a sectores I son todos macrobloques de tipo I.

**Cuadro 7-8 – Tipos de macrobloques de sectores I**

<b>mb_type</b>	<b>Nombre de mb_type</b>	<b>MbPartPredMode ( mb_type, 0 )</b>	<b>Intra16x16PredMode</b>	<b>CodedBlockPatternChroma</b>	<b>CodedBlockPatternLuma</b>
0	I_4x4	Intra_4x4	na	Ecuación 7-22	Ecuación 7-22
1	I_16x16_0_0_0	Intra_16x16	0	0	0
2	I_16x16_1_0_0	Intra_16x16	1	0	0
3	I_16x16_2_0_0	Intra_16x16	2	0	0
4	I_16x16_3_0_0	Intra_16x16	3	0	0
5	I_16x16_0_1_0	Intra_16x16	0	1	0
6	I_16x16_1_1_0	Intra_16x16	1	1	0
7	I_16x16_2_1_0	Intra_16x16	2	1	0
8	I_16x16_3_1_0	Intra_16x16	3	1	0
9	I_16x16_0_2_0	Intra_16x16	0	2	0
10	I_16x16_1_2_0	Intra_16x16	1	2	0
11	I_16x16_2_2_0	Intra_16x16	2	2	0
12	I_16x16_3_2_0	Intra_16x16	3	2	0
13	I_16x16_0_0_1	Intra_16x16	0	0	15
14	I_16x16_1_0_1	Intra_16x16	1	0	15
15	I_16x16_2_0_1	Intra_16x16	2	0	15
16	I_16x16_3_0_1	Intra_16x16	3	0	15
17	I_16x16_0_1_1	Intra_16x16	0	1	15
18	I_16x16_1_1_1	Intra_16x16	1	1	15
19	I_16x16_2_1_1	Intra_16x16	2	1	15
20	I_16x16_3_1_1	Intra_16x16	3	1	15
21	I_16x16_0_2_1	Intra_16x16	0	2	15
22	I_16x16_1_2_1	Intra_16x16	1	2	15
23	I_16x16_2_2_1	Intra_16x16	2	2	15
24	I_16x16_3_2_1	Intra_16x16	3	2	15
25	I_PCM	na	na	na	na

A los tipos de macrobloque del cuadro 7-8 se les asigna la siguiente semántica:

I\_4x4: el macrobloque está codificado como un macrobloque de predicción Intra\_4x4.

I\_16x16\_0\_0\_0, I\_16x16\_1\_0\_0, I\_16x16\_2\_0\_0, I\_16x16\_3\_0\_0, I\_16x16\_0\_1\_0, I\_16x16\_1\_1\_0, I\_16x16\_2\_1\_0, I\_16x16\_3\_1\_0, I\_16x16\_0\_2\_0, I\_16x16\_1\_2\_0, I\_16x16\_2\_2\_0, I\_16x16\_3\_2\_0, I\_16x16\_0\_0\_1, I\_16x16\_1\_0\_1, I\_16x16\_2\_0\_1, I\_16x16\_3\_0\_1, I\_16x16\_0\_1\_1, I\_16x16\_1\_1\_1, I\_16x16\_2\_1\_1, I\_16x16\_3\_1\_1,

I\_16x16\_0\_2\_1, I\_16x16\_1\_2\_1, I\_16x16\_2\_2\_1, I\_16x16\_3\_2\_1: el macrobloque está codificado como un macrobloque de modo predicción Intra\_16x16.

A cada macrobloque de predicción Intra\_16x16 se le asigna un Intra16x16PredMode, el cual especifica el modo de predicción Intra\_16x16. CodedBlockPatternChroma contiene el valor del patrón de bloque de código para la componente croma según se especifica en el cuadro 7-12. CodedBlockPatternLuma indica si hay niveles de coeficiente de transformada c.a. distintos de cero para la componente luma. CodedBlockPatternLuma igual a 0 indica que todos los coeficientes de transformada c.a. en la componente luma del macrobloque son iguales a cero. CodedBlockPatternLuma igual a 15 indica que hay al menos un nivel de coeficientes de transformada c.a. en la componente luma del macrobloques diferente de cero y que requiere un barrido de los niveles de coeficiente de transformada c.a. para los 16 bloques 4x4 en el bloque 16x16.

Intra\_4x4 especifica el modo de predicción de macrobloque e indica que se llama al proceso de predicción Intra\_4x4 descrito en la subcláusula 8.3.1. Intra\_4x4 es un modo de predicción intra macrobloque.

Intra\_16x16 especifica el modo de predicción del macrobloque e indica que se llama al proceso de predicción Intra\_16x16 descrito en la subcláusula 8.3.2. Intra\_16x16 es un modo de predicción intra macrobloque.

Para los macrobloques codificados con mb\_type igual a I\_PCM, se supondrá que el modo de predicción de macrobloque es intra.

En el cuadro 7-9 se especifica el tipo de macrobloque que puede aparecer denominado como macrobloque de tipo SI.

Los tipos de macrobloque correspondientes a los sectores SI se especifican en los cuadros 7-9 y 7-8. El tipo de macrobloque correspondiente al valor 0 de mb\_type se especifica en el cuadro 7-9 y los correspondientes a los valores de mb-type comprendidos entre 1 y 26 se especifican en el cuadro 7-8, en el que hay que restar 1 al valor de mb\_type.

**Cuadro 7-9 – Tipo de macrobloque con valor 0 correspondiente a sectores SI**

mb_type	Nombre de mb_type	MbPartPredMode (mb_type, 0)	Intra16x16PredMode	CodedBlockPatternChroma	CodedBlockPatternLuma
0	SI	Intra_4x4	na	Ecuación 7-22	Ecuación 7-22

La semántica del tipo de macrobloque descrito en el cuadro 7-9 es la siguiente. El macrobloque SI se codifica como macrobloque de predicción Intra\_4x4.

En el cuadro 7-10 se especifica el tipo de macrobloque que puede aparecer denominado genéricamente como macrobloque de tipo P.

Los tipos de macrobloque correspondiente a los sectores P y SP se especifican en los cuadros 7-10 y 7-8. El tipo de macrobloque correspondiente a valores de mb\_type entre 0 y 4 se especifica en el

cuadro 7-10 y para valores de mb\_type entre 5 y 30 se especifica en el cuadro 7-8, en el que hay que restar 5 al valor de mb\_type.

**Cuadro 7-10 – Tipos de macrobloques de valores entre 0 a 4 correspondiente a sectores P y SP**

mb_type	Nombre de mb_type	NumMbPart ( mb_type )	MbPartPredMode ( mb_type, 0 )	MbPartPredMode ( mb_type, 1 )	MbPartWidth ( mb_type )	MbPartHeight ( mb_type )
0	P_L0_16x16	1	Pred_L0	na	16	16
1	P_L0_L0_16x8	2	Pred_L0	Pred_L0	16	8
2	P_L0_L0_8x16	2	Pred_L0	Pred_L0	8	16
3	P_8x8	4	na	na	8	8
4	P_8x8ref0	4	na	na	8	8
se infiere	P_Skip	1	Pred_L0	na	16	16

A continuación se describe la semántica de los tipos de macrobloque del cuadro 7-10.

- P\_L0\_16x16: las muestras del macrobloque se predicen con una partición macrobloque luma de tamaño 16x16 muestras luma y sus correspondientes muestras croma.
- P\_L0\_L0\_MxN, siendo MxN 16x8 u 8x16: las muestras del macrobloque se predicen utilizando dos particiones luma de tamaño MxN igual a 16x8, o dos particiones luma de tamaño MxN igual a 8x16, y sus correspondientes muestras croma, respectivamente.
- P\_8x8: para cada submacrobloque hay un elemento sintáctico adicional (sub\_mb\_type) en el tren de bits que especifica el tipo de submacrobloque correspondiente (véase la subcláusula 7.4.5.2).
- P\_8x8ref0: tiene la misma semántica que P\_8x8 pero el tren de bits no contiene elementos sintácticos para índice de referencia (ref\_idx\_10) y además se supondrá que el valor de ref\_idx\_10[ mbPartIdx ] es 0 para todos los sub-macrobloques del macrobloque (con índices mbPartIdx igual a 0..3).
- P\_Skip: no hay más datos correspondientes al macrobloque en el tren de bits.

La semántica de los modos de predicción de macrobloque (MbPartPredMode( )) del cuadro 7-10 es la siguiente.

- Pred\_L0: especifica que se llama al proceso de interpredicción utilizando la predicción de lista 0. Pred\_L0 es un modo de predicción de macrobloque Inter.

En el cuadro 7-11 se especifican los tipos de macrobloque que pueden aparecer denominados genéricamente como macrobloques de tipo B.

Los tipos de macrobloques correspondientes a sectores B se especifican en los cuadros 7-11 y 7-8. Los valores de 0 a 22 de mb\_type se especifican en el cuadro 7-11 y los valores 23 a 48 de mb\_type se especifican en el cuadro 7-8, en el que hay que restar 23 del valor de mb\_type.

**Cuadro 7-11 – Tipo de macrobloque de valores 0 a 22 correspondiente a sectores B**

mb_type	Nombre del mb_type	NumMbPart ( mb_type )	MbPartPredMode ( mb_type, 0 )	MbPartPredMode ( mb_type, 1 )	MbPartWidth ( mb_type )	MbPartHeight ( mb_type )
0	B_Direct_16x16	na	Directo	na	8	8
1	B_L0_16x16	1	Pred_L0	na	16	16
2	B_L1_16x16	1	Pred_L1	na	16	16
3	B_Bi_16x16	1	BiPred	na	16	16
4	B_L0_L0_16x8	2	Pred_L0	Pred_L0	16	8
5	B_L0_L0_8x16	2	Pred_L0	Pred_L0	8	16
6	B_L1_L1_16x8	2	Pred_L1	Pred_L1	16	8
7	B_L1_L1_8x16	2	Pred_L1	Pred_L1	8	16
8	B_L0_L1_16x8	2	Pred_L0	Pred_L1	16	8
9	B_L0_L1_8x16	2	Pred_L0	Pred_L1	8	16
10	B_L1_L0_16x8	2	Pred_L1	Pred_L0	16	8
11	B_L1_L0_8x16	2	Pred_L1	Pred_L0	8	16
12	B_L0_Bi_16x8	2	Pred_L0	BiPred	16	8
13	B_L0_Bi_8x16	2	Pred_L0	BiPred	8	16
14	B_L1_Bi_16x8	2	Pred_L1	BiPred	16	8
15	B_L1_Bi_8x16	2	Pred_L1	BiPred	8	16
16	B_Bi_L0_16x8	2	BiPred	Pred_L0	16	8
17	B_Bi_L0_8x16	2	BiPred	Pred_L0	8	16
18	B_Bi_L1_16x8	2	BiPred	Pred_L1	16	8
19	B_Bi_L1_8x16	2	BiPred	Pred_L1	8	16
20	B_Bi_Bi_16x8	2	BiPred	BiPred	16	8
21	B_Bi_Bi_8x16	2	BiPred	BiPred	8	16
22	B_8x8	4	na	na	8	8
se infiere	B_Skip	na	Directo	na	8	8

La semántica de los tipos de macrobloque del cuadro 7-11 es la siguiente:

- B\_Direct\_16x16: el tren de bits no contiene diferencias de vector de movimiento o índices de referencia del macrobloque. Las funciones MbPartWidth(B\_Direct\_16x16), y MbPartHeight(B\_Direct\_16x16) se utilizan en el proceso de cálculo de lectores de movimiento y de índices de cuadro descrito en la subcláusula 8.4.1 para predicción de modo directo.
- B\_X\_16x16, siendo X L0, L1 o Bi: las muestras de macrobloque se predicen con una partición macrobloque luma de tamaño 16x16 muestras luma y sus correspondientes muestras croma. Para los macrobloques de tipo B\_X\_16x16, siendo X L0 o L1, el tren de bits contiene una diferencia de vector de movimiento y un índice de referencia del macrobloque. Para los macrobloques de tipo B\_X\_16x16, siendo X Bi, el tren de bits contiene dos diferencias de lectores de movimiento y dos índices de referencia del macrobloque.



- B\_X0\_X1\_MxN, donde X0, X1 hacen referencia a la primera y segunda partición del macrobloque y se sustituyen por L0, L1, o Bi, y siendo MxN 16x8 u 8x16: las muestras del macrobloque se predicen utilizando dos particiones luma de tamaño MxN igual a 16x8, o dos particiones luma de tamaño MxN igual a 8x16, y sus correspondientes muestras croma, respectivamente. Para las particiones macrobloque X0 o X1, siendo X0 o X1 L0 o L1, el tren de bits contiene una diferencia de vector de movimiento y un índice de referencia. Para las particiones macrobloque X0 o X1, siendo X0 o X1 Bi, el tren de bits contiene dos diferencias de vector de movimiento y dos índices de referencia de la partición de macrobloque.
- B\_8x8: para cada submacrobloque hay un elemento sintáctico adicional (sub\_mb\_type) en el tren de bits que especifica el tipo de submacrobloque correspondiente (véase la subcláusula 7.4.5.2).
- B\_Skip: no hay más datos correspondientes al macrobloque en el tren de bits. Las funciones MbPartWidth( B\_Skip ), y MbPartHeight( B\_Skip ) se utilizan en el proceso de cálculo de vectores de movimiento e índices de cuadro de referencia descrito en la subcláusula 8.4.1 para la predicción del modo directo.

La semántica de los modos de predicción de macrobloque (MbPartPredMode( )) del cuadro 7-11 es la siguiente.

- Directo: el tren de bits no contiene diferencias de vector de movimiento o índices de referencia del macrobloque (en caso de B\_Skip o B\_Direct\_16x16). El modo directo es un modo de predicción inter macrobloque.
- Pred\_L0: véase semántica del cuadro 7-10.
- Pred\_L1: indica que se llama al proceso de predicción inter utilizando predicción de lista 1. Pred\_L1 es un modo de predicción inter macrobloque.
- BiPred: indica que se llama al proceso de predicción inter utilizando predicción de lista 0 y lista 1. BiPred es un modo de predicción inter macrobloque.

**pcm\_alignment\_zero\_bit** es un bit igual a 0.

**pcm\_byte[ i ]** es un valor muestra. pcm\_byte[ i ] será distinto de 0. Los primeros 256 valores de pcm\_byte[ i ] representan los valores de las muestras luma en el orden de barrido por filas dentro del macrobloque. Los siguientes  $(256 * (\text{ChromaFormatFactor} - 1)) / 2$  pcm\_byte[ i ] valores representan los valores de las muestras Cb en el orden de barrido por filas dentro del macrobloque. Los últimos  $(256 * (\text{ChromaFormatFactor} - 1)) / 2$  pcm\_byte[ i ] valores representan los valores de muestra Cr en el orden de barrido por filas dentro del macrobloque.

**coded\_block\_pattern** especifica qué bloques de los seis bloques 8x8 – luma y croma – pueden contener niveles de coeficiente transformada distinta de cero. Para macrobloques con modo de predicción distinto de Intra\_16x16, el tren de bits contendría coded\_block\_pattern y las variables CodedBlockPatternLuma y CodedBlockPatternChroma se calculan del modo siguiente:

$$\begin{aligned} \text{CodedBlockPatternLuma} &= \text{coded\_block\_pattern} \% 16 \\ \text{CodedBlockPatternChroma} &= \text{coded\_block\_pattern} / 16 \end{aligned} \quad (7-22)$$

Cuando está presente coded\_block\_pattern, CodedBlockPatternLuma especifica, para cada uno de los bloques luma 8x8 del macrobloque, uno de los siguientes casos.

- Todos los niveles de coeficiente de transformada de los cuatro bloques luma 4x4 del bloque luma 8x8 son iguales a cero.
- Uno o más de los niveles de coeficientes de transformada de uno o más de los bloques luma 4x4 en el bloque luma 8x8 tendrán valores diferentes de cero.

El significado de CodedBlockPatternChroma se especifica en el cuadro 7-12.

**Cuadro 7-12 – Especificación de los valores de CodedBlockPatternChroma**

CodedBlockPatternChroma	Descripción
0	Todos los niveles de coeficiente de transformada croma son iguales a 0.
1	Al menos uno de los niveles de coeficientes transformada c.c. croma tendrá un valor distinto de 0 Todos los niveles c.a. de coeficientes de transformada croma son iguales a 0.
2	Puede haber niveles de coeficientes de transformada c.c. croma distintos de cero. Al menos un nivel de coeficientes de transformada c.a. croma tendrá un valor distinto de cero.

**mb\_qp\_delta** puede cambiar el valor de  $QP_Y$  en la capa macrobloque. El valor decodificado de **mb\_qp\_delta** estará entre  $-26$  y  $+25$ , inclusive. **mb\_qp\_delta** se supondrá igual a 0 cuando no aparezca en ningún otro marco bloque (incluidos los macrobloques de tipos **P\_Skip** y **B\_Skip**).

El valor de  $QP_Y$  se calcula mediante la expresión:

$$QP_Y = (QP_{Y,PREV} + mb\_qp\_delta + 52) \% 52 \quad (7-23)$$

siendo  $QP_{Y,PREV}$  el parámetro de cuantificación luma,  $QP_Y$ , del macrobloque previo en orden de decodificación del sector considerado. Al primer macrobloque del sector,  $QP_{Y,PREV}$  se le asigna inicialmente el valor de  $SliceQP_Y$  calculado mediante la ecuación 7-16 al principio de cada sector.

#### 7.4.5.1 Semántica de predicción de macrobloques

Todas las muestras del macrobloque se predicen. Los modos de predicción se obtienen mediante los siguientes elementos sintácticos.

**prev\_intra4x4\_pred\_mode\_flag**[ luma4x4BlkIdx ] y **rem\_intra4x4\_pred\_mode**[ luma4x4BlkIdx ] especifican la predicción Intra\_4x4 de bloques luma 4x4 con índice luma4x4BlkIdx = 0..15.

**intra\_chroma\_pred\_mode** especifica el tipo de predicción espacial utilizado para croma en macrobloques que utilizan predicción intra 4x4 o intra 16x16, como se muestra en el cuadro 7-13.

**Cuadro 7-13 – Relación entre los modos de predicción Intra\_chroma\_pred\_mode y espacial**

intra_chroma_pred_mode	Modo de predicción croma Intra
0	c.c.
1	Horizontal
2	Vertical
3	Plano

**ref\_idx\_10**[ mbPartIdx ], si lo hubiere, especifica el índice de lista 0 de la imagen de referencia que se utilizará para la predicción.

Los valores posibles de **ref\_idx\_10**[ mbPartIdx ], el índice en la lista 0 de la imagen de referencia y, si procede, la paridad del campo dentro de la imagen de referencia utilizada para la predicción se especifica del modo siguiente:

- si **MbaffFrameFlag** es igual a 0 o **mb\_field\_decoding\_flag** es igual a 0, el valor de **ref\_idx\_10**[ mbPartIdx ] estará entre 0 y **num\_ref\_idx\_10\_active\_minus1**, inclusive;

- de lo contrario (MbaffFrameFlag es igual a 1 y mb\_field\_decoding\_flag es igual a 1) el valor de ref\_idx\_10[ mbPartIdx ] estará entre 0 y 2 \* num\_ref\_idx\_10\_active\_minus1 + 1, inclusive.

Cuando sólo se utilice una imagen de referencia para la predicción inter, los valores ref\_idx\_10[ mbPartIdx ] se supondrán iguales a 0.

**ref\_idx\_11[ mbPartIdx ]** tiene la misma semántica que ref\_idx\_10, sustituyendo l0 y lista 0 por l1 y lista 1, respectivamente.

**mvd\_10[ mbPartIdx ][ 0 ][ compIdx ]** especifica la diferencia entre una componente vectorial que se utilizará y su valor predicho. El índice mbPartIdx especifica a qué participación de macrobloques se asigna mvd\_10. La partición del macrobloque se especifica por mb\_type. La diferencia de la componente del vector movimiento horizontal se decodifica en primer lugar, en el orden de decodificación, y se asigna CompIdx = 0. La componente de vector de movimiento vertical se decodifica en segundo lugar, en el orden de decodificación, y se asigna CompIdx = 1. Los valores posibles de las componentes de mvd\_10[ mbPartIdx ][ 0 ][ compIdx ] se especifican mediante restricciones que se aplican a los valores de variable vector de movimiento que se obtienen a partir de esas componentes, como se describe en el anexo A.

**mvd\_11[ mbPartIdx ][ 0 ][ compIdx ]** tiene la misma semántica que mvd\_10, sustituyendo l0 y L0 por l1 y L1, respectivamente.

#### 7.4.5.2 Semántica de predicción de submacrobloques

**sub\_mb\_type[ mbPartIdx ]** especifica los tipos de submacrobloque.

En los cuadros y la semántica que figuran a continuación se especifican los diversos tipos de submacrobloques correspondientes a sectores P, SP, y B. Cada cuadro presenta el valor de sub\_mb\_type, el nombre de sub\_mb\_type, el número de participaciones submacrobloque utilizadas (dado por la función NumSubMbPart( sub\_mb\_type )), y el modo predicción del submacrobloque (dado por la función SubMbPredMode( sub\_mb\_type )). sub\_mb\_type puede aparecer denominado en el texto como "tipo de submacrobloque" y SubMbPredMode( ) como "modo de predicción de submacrobloque"

Los tipos de submacrobloque correspondientes a los macrobloques de tipo P se especifican en el cuadro 7-14.

**Cuadro 7-14 – Tipos de submacrobloque en macrobloques P**

sub_mb_type[ mbPartIdx ]	Nombre de sub_mb_type[ mbPartIdx ]	NumSubMbPart ( sub_mb_type[ mbPartIdx ] )	SubMbPredMode ( sub_mb_type[ mbPartIdx ] )	SubMbPartWidth ( sub_mb_type[ mbPartIdx ] )	SubMbPartHeight ( sub_mb_type[ mbPartIdx ] )
0	P_L0_8x8	1	Pred_L0	8	8
1	P_L0_8x4	2	Pred_L0	8	4
2	P_L0_4x8	2	Pred_L0	4	8
3	P_L0_4x4	4	Pred_L0	4	4

La semántica de los tipos de submacrobloque del cuadro 7-14, es la siguiente.

- P\_L0\_MxN, siendo MxN 8x8, 8x4, 4x8, ó 4x4: las muestras del submacrobloque se predicen utilizando una partición luma de tamaño MxN igual a 8x8, dos particiones luma de tamaño MxN igual a 8x4, dos particiones luma de tamaño MxN igual a 4x8 o cuatro particiones luma de tamaño MxN igual a 4x4, y sus correspondientes muestras cromas respectivamente.

La semántica de los modos de predicción de submacrobloques (SubMbPredMode()) del cuadro 7-14 es la siguiente:

- Pred\_L0: véase semántica del cuadro 7-10.

En el cuadro 7-15 se especifican los tipos de submacrobloque correspondientes a macrobloques de tipo B.

**Cuadro 7-15 – Tipos de submacrobloque en macrobloques B**

sub_mb_type mbPartIdx	Name of sub_mb_type mbPartIdx	NumSubMbPart (sub_mb_type mbPartIdx  )	SubMbPredMode (sub_mb_type mbPartIdx  )	SubMbPartWidth (sub_mb_type mbPartIdx  )	SubMbPartHeight (sub_mb_type mbPartIdx  )
na	B_Skip	na	Directo	4	4
na	B_Direct_16x16	na	Directo	4	4
0	B_Direct_8x8	na	Directo	4	4
1	B_L0_8x8	1	Pred_L0	8	8
2	B_L1_8x8	1	Pred_L1	8	8
3	B_Bi_8x8	1	BiPred	8	8
4	B_L0_8x4	2	Pred_L0	8	4
5	B_L0_4x8	2	Pred_L0	4	8
6	B_L1_8x4	2	Pred_L1	8	4
7	B_L1_4x8	2	Pred_L1	4	8
8	B_Bi_8x4	2	BiPred	8	4
9	B_Bi_4x8	2	BiPred	4	8
10	B_L0_4x4	4	Pred_L0	4	4
11	B_L1_4x4	4	Pred_L1	4	4
12	B_Bi_4x4	4	BiPred	4	4

La semántica de los tipos de macrobloque del cuadro 7-15 es la siguiente:

- B\_Skip y B\_Direct\_16x16: el tren de bits no contiene diferencias de vector de movimiento ni índices de referencia correspondientes al submacrobloque. Las funciones SubMbPartWidth() y SubMbPartHeight() se utilizan en el proceso de cálculo de vectores de movimiento e índices de cuadro de referencia descrito en la subcláusula 8.4.1 para la predicción de modo directo.
- B\_Direct\_8x8: el tren de bits no contiene diferencias de vector de movimiento o índices de referencia correspondientes al submacrobloque. Las funciones SubMbPartWidth( B\_Direct\_8x8 ) y SubMbPartHeight( B\_Direct\_8x8 ) se utilizan en el

proceso de cálculo de vectores de movimiento e índices de cuadro de referencia descrito en la subcláusula 8.4.1 para la predicción de modo directo.

- B\_X\_MxN, siendo X L0, L1, o Bi, y siendo MxN 8x8, 8x4, 4x8 ó 4x4: las muestras del submacrobloque se predicen utilizando una partición luma de tamaño MxN igual a 8x8, o utilizando dos particiones luma de tamaño MxN igual a 8x4, o dos particiones luma de tamaño MxN igual a 4x8 o cuatro particiones luma de tamaño MxN igual a 4x4, y sus correspondientes muestras croma, respectivamente. Todas las particiones submacrobloque tienen el mismo índice de referencia. Para las particiones submacrobloque MxN de un submacrobloque con sub\_mb\_type igual a B\_X\_MxN, siendo X L0 o L1, el tren de bits contiene una diferencia de vector de movimiento. En las particiones submacrobloques MxN de un submacrobloque con sub\_mb\_type igual a B\_Bi\_MxN, el tren de bits contiene dos diferencias de vector de movimiento.

La semántica de los modos de predicción de submacrobloque (SubMbPredMode( )) del cuadro 7-15 es la siguiente:

- Directo: véase semántica del cuadro 7-11.
- Pred\_L0: véase semántica del cuadro 7-10.
- Pred\_L1: véase semántica del cuadro 7-11.
- BiPred: véase semántica del cuadro 7-11.

**ref\_idx\_10**[ mbPartIdx ] tiene la misma semántica que ref\_idx\_10 de la subcláusula 7.4.5.1.

**ref\_idx\_11**[ mbPartIdx ] tiene la misma semántica que ref\_idx\_11 de la subcláusula 7.4.5.1.

**mvd\_10**[ mbPartIdx ][ subMbPartIdx ][ compIdx ] tiene la misma semántica que mvd\_10 de la subcláusula 7.4.5.1, con la única diferencia de que se aplica al índice de partición submacrobloque con subMbPartIdx. Los índices mbPartIdx y subMbPartIdx especifican a qué partición macrobloque o partición submacrobloque se asigna mvd\_10.

**mvd\_11**[ mbPartIdx ][ subMbPartIdx ][ compIdx ] tiene la misma semántica que mvd\_11 de la subcláusula 7.4.5.1.

### 7.4.5.3 Semántica de datos residuales

La estructura sintáctica residual\_block( ), que se utiliza para el análisis sintáctico de los niveles de coeficientes de transformada, se define del modo siguiente:

- Si entropy\_coding\_mode\_flag es igual a 0, a residual\_block se le asigna el valor de residual\_block\_cavlc, que se utiliza para el análisis sintáctico de elementos sintácticos de los niveles de coeficientes de transformada.
- De lo contrario (entropy\_coding\_mode\_flag es igual a 1), a residual\_block se le asigna el valor de residual\_block\_cabac, que se utiliza para el análisis sintáctico de elementos sintácticos de los niveles de coeficiente de transformada.

En función de mb\_type, luma o croma, la estructura sintáctica residual\_block( coeffLevel, maxNumCoeff ) se utiliza con los argumentos coeffLevel, que es una lista que contiene el maxNumCoeff de niveles de coeficiente de transformada analizados mediante residual\_block( ), y maxNumCoeff del modo siguiente:

- En función de  $\text{MbPartPredMode}(\text{mb\_type}, 0)$ , se aplica lo siguiente.
  - Si  $\text{MbPartPredMode}(\text{mb\_type}, 0)$  es igual a  $\text{Intra\_16x16}$ , los niveles de coeficientes de transformada se analizan sintácticamente en la lista  $\text{Intra16x16DCLevel}$  y en las 16 listas  $\text{Intra16x16ACLevel}[i]$ .  $\text{Intra16x16DCLevel}$  contiene los 16 niveles de coeficientes de transformada de los niveles de coeficientes de transformada c.c. para cada bloque luma  $4x4$ . Para cada uno de los 16 bloques luma  $4x4$ , indexados mediante  $i = 0..15$ , se analizan los 15 niveles de coeficientes de transformada c.a. del  $i$ -ésimo bloque en la  $i$ -ésima lista  $\text{Intra16x16ACLevel}[i]$ .
  - De lo contrario ( $\text{MbPartPredMode}(\text{mb\_type}, 0)$  es distinto de  $\text{Intra\_16x16}$ ), para cada uno de los 16 bloques luma  $4x4$ , indexados mediante  $i = 0..15$ , se analizan los 16 niveles de coeficientes de transformada del  $i$ -ésimo bloque en la  $i$ -ésima lista  $\text{LumaLevel}[i]$ .
- Para cada componente croma, indexada mediante  $i\text{CbCr} = 0..1$ , se analizan los 4 niveles de coeficientes de transformada c.c. de los bloques croma  $4x4$  en la  $i\text{CbCr}$ -ésima lista  $\text{ChromaDCLevel}[i\text{CbCr}]$ .
- Para cada bloque croma  $4x4$  indexado mediante  $i4x4 = 0..3$ , de cada componente croma, indexada mediante  $i\text{CbCr} = 0..1$ , se analizan los 15 niveles de coeficientes de transformada AC en la  $i4x4$ -ésima lista de la  $i\text{CbCr}$ -ésima componente  $\text{ChromaACLevel}[i\text{CbCr}][i4x4]$ .

#### 7.4.5.3.1 Semántica de la CAVLC de bloques residuales

La función  $\text{TotalCoeff}(\text{coeff\_token})$  que se utiliza en la subcláusula 7.3.5.3.1 devuelve el número de coeficientes de transformada distintos de cero calculados a partir de  $\text{coeff\_token}$ .

La función  $\text{TrailingOnes}(\text{coeff\_token})$  que se utiliza en la subcláusula 7.3.5.3.1 devuelve los unos de cola calculados a partir de  $\text{coeff\_token}$ .

**coeff\_token** especifica el número total de niveles de coeficientes de transformada distintos de cero y el número total de niveles de coeficientes de transformada de unos de cola en un barrido de niveles de coeficientes de transformada. El nivel de coeficiente de transformada de unos de cola es igual a uno, dos o tres niveles de coeficientes de transformada distintos de cero consecutivos cuyo valor absoluto es igual a uno al final de un barrido de niveles de coeficientes de transformada distintos de cero. Los valores posibles de  $\text{coeff\_token}$  se especifican en la subcláusula 9.2.1.

**trailing\_ones\_sign\_flag** especifica el signo de un nivel de coeficientes de transformada de unos de cola, del modo siguiente:

- Si  $\text{trailing\_ones\_sign\_flag}$  es igual a 0, el correspondiente nivel de coeficientes de transformada se decodifica como +1.
- De lo contrario ( $\text{trailing\_ones\_sign\_flag}$  igual a 1), el correspondiente nivel de coeficientes de transformada se decodifica como -1.

**level\_prefix** y **level\_suffix** especifica el valor de un nivel de coeficiente de transformada distintos de cero. Los valores posibles de  $\text{level\_prefix}$  y  $\text{level\_suffix}$  se especifican en la subcláusula 9.2.2.

**total\_zeros** especifica el número total de niveles de coeficiente de transformada de valor cero que están antes de la posición del último nivel de coeficiente de transformada distinto de cero en un barrido de niveles de coeficiente de transformada. Los valores posibles de  $\text{total\_zeros}$  se especifican en la cláusula 9.2.3.

**run\_before** especifica el número de niveles de coeficientes de transformada consecutivos de valor cero en el barrido y que aparecen antes de un nivel de coeficiente de transformada de valor distinto de cero. Los valores posibles de  $\text{run\_before}$  se especifica en la subcláusula 9.2.3.

$\text{coeffLevel}$  contiene el  $\text{maxNumCoeff}$  de niveles de coeficientes de transformada correspondientes a la lista considerada de niveles de coeficientes de transformada.

### 7.4.5.3.2 Semántica de la CABAC de bloques residuales

**coded\_block\_flag** indica si el bloque contiene niveles de coeficientes de transformada distintos de cero, del modo siguiente:

- Si **coded\_block\_flag** es igual a 0, el bloque no contiene niveles de coeficientes transformada distintos de cero.
- De lo contrario (**coded\_block\_flag** es igual a 1), el bloque contiene al menos un nivel de coeficientes de transformada distinto de cero.

**significant\_coeff\_flag[ i ]** indica si el nivel de coeficientes de transformada en la posición de barrido *i* es distinto de cero, del modo siguiente:

- Si **significant\_coeff\_flag[ i ]** es igual a 0, el nivel de coeficientes de transformada en la posición de barrido *i* es igual a 0.
- De lo contrario (**significant\_coeff\_flag[ i ]** es igual a 1), el nivel de coeficiente de transformada en la posición de barrido *i* tiene un valor distinto de cero.

**last\_significant\_coeff\_flag[ i ]** indica si para una posición de barrido *i* hay niveles de coeficientes de transformada distintos a cero en las subsiguientes posiciones de barrido, desde *i* + 1 hasta **maxNumCoeff** – 1, del modo siguiente:

- Si **last\_significant\_coeff\_flag[ i ]** es igual a 1, todos los niveles de coeficientes de transformada siguientes (en orden de barrido) del bloque tienen un valor igual a 0.
- De lo contrario (**last\_significant\_coeff\_flag[ i ]** es igual a 0), aún hay más niveles de coeficientes de transformada distintos de cero en el trayecto de barrido.

**coeff\_abs\_level\_minus1[ i ]** es el valor absoluto de nivel de coeficiente de transformada menos 1. Los valores posibles de **coeff\_abs\_level\_minus1** están limitados por las restricciones de la subcláusula 8.5.

**coeff\_sign\_flag[ i ]** indica el signo del nivel de coeficiente de transformada, del modo siguiente:

- Si **coeff\_sign\_flag** es igual a 0, el correspondiente nivel de coeficientes de transformada tiene un valor positivo.
- De lo contrario (**coeff\_sign\_flag** es igual a 1) el correspondiente nivel de coeficientes de transformada tiene un valor negativo.

**coeffLevel** contiene **maxNumCoeff** niveles de coeficientes de transformada correspondientes a la lista considerada de niveles de coeficiente de transformada.

## 8 Proceso de decodificación

Los resultados de aplicar este proceso son muestras decodificadas de la imagen considerada (a la cual se hace referencia a veces mediante la variable **CurrPic**).

En esta cláusula se describe el proceso de decodificación utilizando los elementos sintácticos y las variables en mayúsculas de la cláusula 7.

El proceso de decodificación está especificado para que todos los decodificadores produzcan resultados numéricos idénticos. Los procesos de decodificación que generen resultados idénticos al proceso descrito en este documento son conformes con los requisitos del proceso de codificación de esta Recomendación | Norma Internacional.

En esta cláusula por imagen se entiende una imagen primaria, y por sector se entiende un sector de una imagen primaria. De igual modo, por partición de datos de sector se sobreentiende una partición de datos de sector de una imagen primaria.

A continuación se resume el proceso de codificación.

- La decodificación de unidades NAL se describe en la subcláusula 8.1.

- En la subcláusula 8.2 se describen los procesos de decodificación utilizando los elementos sintácticos de la capa de sector y capas superiores.
  - Las variables y funciones relacionadas con el número de orden de la imagen se definen en la subcláusula 8.2.1 (sólo es necesario llamarlas para un sector de una imagen).
  - Las variables y funciones relacionadas con el mapeado de macrobloques en grupos de sectores se define en la subcláusula 8.2.2 (sólo es necesario llamarlas para un sector de una imagen).
  - El método de combinación de varias particiones cuando se utilizan particiones de datos de sector se describe en la subcláusula 8.2.3.
  - En la subcláusula 8.2.4 se describe la obtención de listas de imágenes de referencia necesarias para la predicción inter, cálculo que ha de realizarse antes de decodificar cada sector.
  - Si la imagen considerada es una imagen de referencia y ya han sido decodificados todos los sectores de dicha imagen, en la subcláusula 8.2.5 se describe el proceso de marcado de imágenes de referencia decodificadas el cual especifica cómo se utiliza la imagen considerada en el proceso de decodificación para la predicción inter de las siguientes imágenes decodificadas.
- En las subcláusulas 8.3, 8.4, 8.5, 8.6 y 8.7 se describen los procesos de decodificación utilizando los elementos sintácticos de la capa macrobloque y capas superiores.
  - El proceso de predicción intra aplicable a macrobloques I y SI, salvo los macrobloques I\_PCM, descrito en la subcláusula 8.3, da como resultado las muestras de predicción intra. El proceso de reconstrucción de imágenes para macrobloques I\_PCM se especifica directamente en la subcláusula 8.3. El resultado de este proceso son las muestras construidas antes de aplicar el proceso de filtrado de bloques.
  - En la subcláusula 8.4 se describe el proceso de predicción inter aplicable a macrobloques P y B, que genera como resultado las muestras de predicción inter.
  - En la subcláusula 8.5 se describen los procesos de decodificación de coeficientes de transformada y de reconstrucción de imágenes antes de aplicar el proceso de filtrado bloques. Ese proceso calcula las muestras correspondientes a macrobloques I y B y las correspondientes a macrobloques P en sectores P. El resultado son muestras reconstruidas antes de aplicar el proceso de filtrado de bloques.
  - En la subcláusula 8.6 se especifica el proceso de decodificación aplicable a macrobloques P en sectores SP o a macrobloques SI. Ese proceso calcula muestras de macrobloques P en sectores SP y en macrobloques SI. El resultado son muestras reconstruidas antes de aplicar el proceso de filtrado de bloques.
  - A las muestras reconstruidas antes de aplicar el proceso de filtrado bloques que están cerca de los bordes de bloques y macrobloques se les aplica el proceso de filtrado de bloques descrito en la subcláusula 8.7, el cual produce como resultado las muestras decodificadas.

## **8.1 Proceso de decodificación de unidades NAL**

Este proceso acepta como argumentos unidades NAL.

El proceso genera como resultado las estructuras sintácticas RBSP encapsuladas dentro de unidades NAL.

El proceso de decodificación de cada unidad NAL extrae la estructura sintáctica RBSP de la unidad NAL y seguidamente aplica los procesos de decodificación especificados para la estructura sintáctica RBSP de la unidad NAL, según se describe a continuación.



En la subcláusula 8.2 se describe el proceso de decodificación de unidades NAL cuyo `nal_unit_type` está entre 1 y 5.

En la subcláusula 8.3 se describe el proceso de decodificación de un macrobloque o de una parte del mismo codificado en unidades NAL cuyo `nal_unit_type` es igual a 1, 2 y 5.

En la subcláusula 8.4 se describe el proceso de decodificación de un macrobloque o una parte del mismo codificado en unidades NAL cuyo `nal_unit_type` es igual a 1 y 2.

En la subcláusula 8.5 se describe el proceso de decodificación de un macrobloque o de una parte del mismo codificado en unidades NAL cuyo `nal_unit_type` es igual a 1 o está entre 3 y 5.

En la subcláusula 8.6 se describe el proceso de decodificación de un macrobloque o de una parte del mismo codificado en unidades NAL cuyo `nal_unit_type` es igual a 1 o está entre 3 y 5.

En la subcláusula 8.7 se describe el proceso de decodificación de un macrobloque o de una parte del mismo codificado en unidades NAL cuyo `nal_unit_type` está entre 1 y 5.

Las unidades NAL cuyo `nal_unit_type` es igual a 7 y 8 contienen, respectivamente, conjuntos de parámetros secuencia y conjuntos de parámetros imagen. Los conjuntos de parámetros imagen se utilizan en el proceso de decodificación de otras unidades NAL si se indica la referencia a un conjunto de parámetros imagen dentro de las cabeceras de sector de cada imagen. Los conjuntos de parámetros secuencias se utilizan en el proceso de decodificación de otras unidades NAL si se indica la referencia a un conjunto de parámetros secuencia dentro de los conjuntos de parámetros de imagen de cada secuencia.

No existe un proceso de decodificación normativo para unidades NAL del `nal_unit_type` igual a 6, 9, 10, 11 y 12.

## **8.2 Proceso de decodificación de sectores**

### **8.2.1 Proceso de decodificación del número de orden de la imagen**

Este proceso genera como resultado `TopFieldOrderCnt` (si fuera aplicable) y `BottomFieldOrderCnt` (si fuera aplicable).

El número de orden de la imagen se utiliza para determinar la ordenación inicial de imágenes de referencia en la decodificación de sectores B (véanse las subcláusulas 8.2.4.2.3 y 8.2.4.2.4) con objeto de representar diferencias de ordenación de imágenes entre cuadros o entre campos para calcular vectores de movimiento en modo directo temporal (véase la subcláusula 8.4.1.2.3), para la predicción ponderada de modo implícito en sectores B (véase la subcláusula 8.4.2.3.2) y para comprobar la conformidad del decodificador (véase la subcláusula C.4).

La información relativa al número de orden de imágenes se calcula para cada cuadro, cada campo (decodificado a partir de un campo codificado o de una parte de un cuadro decodificado), o cada par de campos complementarios del modo siguiente:

- Cada cuadro codificado tiene asociado dos números de orden de imágenes, denominados `TopFieldOrderCnt` y `BottomFieldOrderCnt` que corresponden, respectivamente, al campo superior y al campo inferior.
- Cada campo codificado tiene asociado un número de orden de imágenes, denominado `TopFieldOrderCnt` para el campo superior codificado, y `BottomFieldOrderCnt` para el campo inferior codificado.
- Cada par de campo complementarios tiene asociado dos números de orden de imágenes, denominados `TopFieldOrderCnt` y `BottomFieldOrderCnt` que corresponden, respectivamente, al campo superior codificado y al campo inferior codificado.

`TopFieldOrderCnt` y `BottomFieldOrderCnt` indican el orden de imágenes del correspondiente campo superior o campo inferior relativo al primer campo resultante de la imagen IDR anterior o de

la imagen de referencia anterior que tenía una `memory_management_control_operation` de valor igual a 5 en orden de decodificación.

Para obtener `TopFieldOrderCnt` y `BottomFieldOrderCnt` se llama a uno de los procesos de decodificación del número de orden de imágenes de tipo 0, 1 y 2, que se describen, respectivamente, en las cláusulas 8.2.1.1, 8.2.1.2 y 8.2.1.3. Si la imagen considerada contiene un `memory_management_control_operation` de valor igual a 5, después de decodificarla el valor de `tempPicOrderCnt` se pone a `PicOrderCnt(CurrPic)`, el valor de `TopFieldOrderCnt` de la imagen considerada (si la hubiere) se pone a `TopFieldOrderCnt - tempPicOrderCnt`, y el valor de `BottomFieldOrderCnt` de la imagen considerada (si la hubiere) se pone a `BottomFieldOrderCnt - tempPicOrderCnt`.

El tren de bits no contendrá datos que causen que el resultado de aplicar `Min( TopFieldOrderCnt, BottomFieldOrderCnt )` sea distinto de 0 para un cuadro IDR codificado, que `TopFieldOrderCnt` sea distinto de 0 para un campo superior IDR codificado o que `BottomFieldOrderCnt` sea distinto de 0 para un campo inferior IDR. Así pues, al menos uno de `TopFieldOrderCnt` y `BottomFieldOrderCnt` será igual a 0 para los campos de un cuadro IDR codificado.

Cuando la imagen considerada no es una imagen IDR, se aplica lo siguiente:

- Sea `listD` la variable que contiene la lista de valores `TopFieldOrderCnt` y `BottomFieldOrderCnt` relacionados con la lista de imágenes y que incluye todo lo siguiente:
  - la primera imagen de la lista es la imagen anterior de cualquiera de los siguientes tipos
    - una imagen IDR;
    - una imagen que contiene una `memory_management_control_operation` de valor 5;
  - las siguientes imágenes adicionales.
    - Si `pic_order_cnt_type` es igual a 0, ninguna de las imágenes que le siguen a la primera imagen de acuerdo con el orden del proceso de decodificación es un cuadro "inexistente", deducido por el proceso de decodificación para vacíos en `frame_num` especificado en 8.2.5.2, y bien preceden la imagen actual o son la imagen actual, de acuerdo con el orden del proceso de decodificación. La imagen actual se incluye en `listD` antes de que se invoque el proceso de marcación de imágenes de referencia decodificadas, si `pic_order_cnt_type` es igual a 0 y la imagen actual no es un cuadro "inexistente" deducido por el proceso de decodificación para vacíos en `frame_num` especificado en 8.2.5.2.
    - De lo contrario (`pic_order_cnt_type` es diferente de 0), todas las demás imágenes que le siguen a la primera imagen de la lista de acuerdo con el orden del proceso de decodificación y que bien anteceden en la imagen actual de acuerdo con el orden de codificación o son la imagen actual. La imagen actual se incluye en `listD` antes de que se invoque el proceso de marcación de imágenes de referencia decodificadas si `pic_order_cnt_type` es diferente de 0.
- Sea `listO` la variable que contiene los elementos de `listD` en orden ascendente. La `listO` no contendrá lo siguiente:
  - un par de `TopFieldOrderCnt` y `BottomFieldOrderCnt` de un cuadro o de un par de campos complementarios cuyas posiciones no son consecutivas en `listO`.
  - dos `TopFieldOrderCnt` con igual valor;
  - dos `BottomFieldOrderCnt` con igual valor;
  - un `BottomFieldOrderCnt` con el mismo valor que un `TopFieldOrderCnt`, a no ser que ambos pertenezcan al mismo cuadro codificado o par de campos complementarios.

El tren de bits no contendrá datos que den lugar a valores de `TopFieldOrderCnt`, `BottomFieldOrderCnt`, `PicOrderCntMsb` o `FrameNumOffset`, utilizados en el proceso de

codificación descrito en las subcláusulas 8.2.1.1 a 8.2.1.3, que superen la gama de valores  $-2^{31}$  a  $2^{31}-1$ , inclusive.

La función  $\text{PicOrderCnt}(\text{picX})$  se define del modo siguiente:

```
if( picX es un cuadro o un par de campos complementarios)
    PicOrderCnt( picX ) = Min( TopFieldOrderCnt, BottomFieldOrderCnt ) del cuadro o par de campos
    complementarios picX
else if( picX es un campo superior )
    PicOrderCnt( picX ) = TopFieldOrderCnt del campo picX
else if( picX es un campo inferior )
    PicOrderCnt( picX ) = BottomFieldOrderCnt del campo picX
```

 (8-1)

La función  $\text{DiffPicOrderCnt}(\text{picA}, \text{picB})$  se define del modo siguiente:

$$\text{DiffPicOrderCnt}(\text{picA}, \text{picB}) = \text{PicOrderCnt}(\text{picA}) - \text{PicOrderCnt}(\text{picB})$$
 (8-2)

El tren de bits sólo contendrá datos que den lugar a valores de  $\text{DiffPicOrderCnt}(\text{picA}, \text{picB})$ , utilizados en el proceso de decodificación, en la gama  $-2^{15}$  y  $2^{15} - 1$ , inclusive.

NOTA – Sea X la imagen considerada y sean Y y Z dos imágenes en la misma secuencia; se considera que Y y Z están en el mismo sentido de orden de salida que X cuando  $\text{DiffPicOrderCnt}(X, Y)$  y  $\text{DiffPicOrderCnt}(X, Z)$  son positivos o ambos son negativos.

NOTA – En muchas aplicaciones se asigna  $\text{PicOrderCnt}(X)$  un valor proporcional al tiempo de muestreo de la imagen X relativa al tiempo de muestreo de una imagen IDR.

Cuando la imagen considerada contiene un `memory_management_control_operation` de valor 5,  $\text{PicOrderCnt}(\text{CurrPic})$  será mayor que  $\text{PicOrderCnt}$  (cualquier otra imagen de la lista D).

### 8.2.1.1 Proceso de decodificación de tipo 0 de cómputo de orden de imágenes

Se llama este proceso cuando `pic_order_cnt_type` es igual a 0.

Este proceso acepta como argumento  $\text{PicOrderCntMsb}$  de la imagen de referencia anterior en el orden de decodificación, según se especifica en esta subcláusula.

Este proceso genera como resultado  $\text{TopFieldOrderCnt}$  o  $\text{BottomFieldOrderCnt}$ , o ambos.

Las variables  $\text{prevPicOrderCntMsb}$  y  $\text{prevPicOrderCntLsb}$  se calculan del modo siguiente:

- Si la imagen considerada es una imagen IDR,  $\text{prevPicOrderCntMsb}$  se pone a 0 y  $\text{prevPicOrderCntLsb}$  se pone a 0.
- De lo contrario (la imagen considerada no es una imagen IDR), se cumple lo siguiente.
  - Si la imagen de referencia anterior en el orden de decodificación contiene una `memory_management_control_operation` de valor igual a 5, se cumple lo siguiente.
    - Si la imagen de referencia anterior en el orden de decodificación no es un campo inferior,  $\text{prevPicOrderCntMsb}$  se pone a 0 y  $\text{prevPicOrderCntLsb}$  se pone igual al valor de  $\text{TopFieldOrderCnt}$  de la imagen de referencia anterior, de acuerdo con el orden de decodificación.
    - De lo contrario (la imagen de referencia anterior en el orden de decodificación es un campo inferior),  $\text{prevPicOrderCntMsb}$  se pone a 0 y  $\text{prevPicOrderCntLsb}$  se pone a 0.
  - De lo contrario (la imagen de referencia anterior en el orden de decodificación no incluía una `memory_management_control_operation` de valor igual a 5), se asigna a  $\text{prevPicOrderCntMsb}$  el valor de  $\text{PicOrderCntMsb}$  de la imagen de referencia anterior en el orden de decodificación y se asigna a  $\text{prevPicOrderCntLsb}$  el valor de `pic_order_cnt_lsb` de la imagen de referencia anterior en orden de decodificación.

PicOrderCntMsb de la imagen considerada se calcula del modo siguiente:

```
if( ( pic_order_cnt_lsb < prevPicOrderCntLsb ) &&
    ( ( prevPicOrderCntLsb - pic_order_cnt_lsb ) >= ( MaxPicOrderCntLsb / 2 ) ) )
    PicOrderCntMsb = prevPicOrderCntMsb + MaxPicOrderCntLsb
else if( ( pic_order_cnt_lsb > prevPicOrderCntLsb ) &&
    ( ( pic_order_cnt_lsb - prevPicOrderCntLsb ) > ( MaxPicOrderCntLsb / 2 ) ) )
    PicOrderCntMsb = prevPicOrderCntMsb - MaxPicOrderCntLsb
else
    PicOrderCntMsb = prevPicOrderCntMsb
```

(8-3)

Cuando la imagen considerada no es un campo inferior, TopFieldOrderCnt se calcula del modo siguiente:

```
if( !field_pic_flag || !bottom_field_flag )
    TopFieldOrderCnt = PicOrderCntMsb + pic_order_cnt_lsb
```

(8-4)

Cuando la imagen considerada no es un campo superior, BottomFieldOrderCnt se calcula del modo siguiente:

```
if( !field_pic_flag )
    BottomFieldOrderCnt = TopFieldOrderCnt + delta_pic_order_cnt_bottom
else if( bottom_field_flag )
    BottomFieldOrderCnt = PicOrderCntMsb + pic_order_cnt_lsb
```

(8-5)

### 8.2.1.2 Proceso de decodificación del número de orden de imágenes, tipo 1

Se llama a este proceso cuando pic\_order\_cnt\_type es igual a 1.

Este proceso acepta como argumento FrameNumOffset de la imagen anterior en orden de decodificación, según se especifica en esta subcláusula.

Este proceso genera como resultado TopFieldOrderCnt o BottomFieldOrderCnt, o ambos.

En esta subcláusula se describe cómo calcular los valores de TopFieldOrderCnt y BottomFieldOrderCnt. Sea prevFrameNum igual a frame\_num de la imagen anterior en orden de decodificación.

Cuando la imagen considerada no es una imagen IDR, la variable prevFrameNumOffset se calcula del modo siguiente:

- Si la imagen anterior en el orden de decodificación incluye una memory\_management\_control\_operation de valor igual a 5, prevFrameNumOffset se pone a 0.
- De lo contrario (la imagen anterior en el orden de decodificación no incluía una memory\_management\_control\_operation de valor igual a 5), se asigna a prevFrameNumOffset el valor de FrameNumOffset de la imagen anterior según el orden de decodificación.

NOTA – Si gaps\_in\_frame\_num\_value\_allowed\_flag es igual a 1, la imagen anterior según el orden de decodificación puede ser un cuadro "inexistente" deducido por el proceso de decodificación para vacíos en frame\_num especificado en la subcláusula 8.2.5.2.

Los pasos para realizar los cálculos son los siguientes:

1. Cálculo de la variable FrameNumOffset:

```
if( nal_unit_type == 5 )
    FrameNumOffset = 0
else if( prevFrameNum > frame_num )
    FrameNumOffset = prevFrameNumOffset + MaxFrameNum
else
    FrameNumOffset = prevFrameNumOffset
```

(8-6)

2. Cálculo de la variable absFrameNum:

```
if( num_ref_frames_in_pic_order_cnt_cycle != 0 )
    absFrameNum = FrameNumOffset + frame_num
else
    absFrameNum = 0
if( nal_ref_idc == 0 && absFrameNum > 0 )
    absFrameNum = absFrameNum - 1
```

 (8-7)

3. Cuando absFrameNum > 0, picOrderCntCycleCnt y frameNumInPicOrderCntCycle se calculan del modo siguiente:

```
if( absFrameNum > 0 ) {
    picOrderCntCycleCnt = ( absFrameNum - 1 ) / num_ref_frames_in_pic_order_cnt_cycle
    frameNumInPicOrderCntCycle = ( absFrameNum - 1 ) % num_ref_frames_in_pic_order_cnt_cycle
}
```

 (8-8)

4. Cálculo de la variable expectedDeltaPerPicOrderCntCycle:

```
expectedDeltaPerPicOrderCntCycle = 0
for( i = 0; i < num_ref_frames_in_pic_order_cnt_cycle; i++ )
    expectedDeltaPerPicOrderCntCycle += offset_for_ref_frame[ i ]
```

 (8-9)

5. Cálculo de la variable expectedPicOrderCnt:

```
if( absFrameNum > 0 ){
    expectedPicOrderCnt = picOrderCntCycleCnt * expectedDeltaPerPicOrderCntCycle
    for( i = 0; i <= frameNumInPicOrderCntCycle; i++ )
        expectedPicOrderCnt = expectedPicOrderCnt + offset_for_ref_frame[ i ]
} else
    expectedPicOrderCnt = 0
if( nal_ref_idc == 0 )
    expectedPicOrderCnt = expectedPicOrderCnt + offset_for_non_ref_pic
```

 (8-10)

6. Cálculo de las variables TopFieldOrderCnt o BottomFieldOrderCnt:

```
if( !field_pic_flag ) {
    TopFieldOrderCnt = expectedPicOrderCnt + delta_pic_order_cnt[ 0 ]
    BottomFieldOrderCnt = TopFieldOrderCnt +
        offset_for_top_to_bottom_field + delta_pic_order_cnt[ 1 ]
} else if( !bottom_field_flag )
    TopFieldOrderCnt = expectedPicOrderCnt + delta_pic_order_cnt[ 0 ]
else
    BottomFieldOrderCnt = expectedPicOrderCnt + offset_for_top_to_bottom_field + delta_pic_order_cnt[ 0 ]
```

 (8-11)

### 8.2.1.3 Proceso de decodificación del número de orden de imágenes, tipo 2

Se llama a este proceso cuando pic\_order\_cnt\_type es igual a 2.

El resultado de este proceso es TopFieldOrderCnt o BottomFieldOrderCnt, o ambos.

Sea prevFrameNum igual al frame\_num de la imagen anterior en el orden de decodificación.

Si la imagen considerada no es una imagen IDR, la variable prevFrameNumOffset se calcula del siguiente modo:

- Si la imagen anterior en el orden de decodificación incluye una memory\_management\_control\_operation de valor 5, prevFrameNumOffset se pone a 0
- De lo contrario (la imagen anterior en el orden de codificación no incluye una memory\_management\_control\_operation de valor 5), el valor de prevFrameNumOffset es el valor de FrameNumOffset de la imagen anterior según el orden de decodificación.

NOTA – Si gaps\_in\_frame\_num\_value\_allowed\_flag es igual a 1, la imagen anterior según el orden de decodificación puede ser un cuadro "no existente" deducido por el proceso de decodificación para vacíos en frame\_num especificado en la subcláusula 8.2.5.2.

Cálculo de la variable FrameNumOffset:

```
if( nal_unit_type == 5 )
    FrameNumOffset = 0
else if( prevFrameNum > frame_num )
    FrameNumOffset = prevFrameNumOffset + MaxFrameNum
else
    FrameNumOffset = prevFrameNumOffset
```

(8-12)

Cálculo de la variable tempPicOrderCnt:

```
if( nal_unit_type == 5 )
    tempPicOrderCnt = 0
else if( nal_ref_idc == 0 )
    tempPicOrderCnt = 2 * ( FrameNumOffset + frame_num ) - 1
else
    tempPicOrderCnt = 2 * ( FrameNumOffset + frame_num )
```

(8-13)

Cálculo de las variables TopFieldOrderCnt o BottomFieldOrderCnt:

```
if( !field_pic_flag ) {
    TopFieldOrderCnt = tempPicOrderCnt
    BottomFieldOrderCnt = tempPicOrderCnt
} else if( bottom_field_flag )
    BottomFieldOrderCnt = tempPicOrderCnt
else
    TopFieldOrderCnt = tempPicOrderCnt
```

(8-14)

NOTA – El tipo 2 de número de orden de imágenes no se puede utilizar en una secuencia de vídeo codificado que contiene imágenes que no son de referencia consecutivas que pudiera dar lugar a que más de una de esas imágenes tuvieran el mismo valor de TopFieldOrderCnt o el mismo valor que BottomFieldOrderCnt.

NOTA – El orden que resulta de decodificar el tipo 2 de número de orden de imágenes es idéntico al orden de decodificación.

## 8.2.2 Proceso de decodificación del mapeado de macrobloques en grupos de sectores

Este proceso acepta como argumentos el conjunto de parámetros de imagen activa y la cabecera del sector que se va a decodificar.

Ese proceso genera como resultado el mapeado de macrobloques en grupos de sectores, MbToSliceGroupMap.

Se llama a este proceso al comienzo de cada sector.

NOTA – Este proceso genera el mismo resultado para todos los sectores de una imagen.

Cuando num\_slice\_groups\_minus1 es igual a 1 y slice\_group\_map\_type es igual a 3, 4, ó 5, el tamaño y forma de los grupos de sectores 0 y 1 está determinado por slice\_group\_change\_direction\_flag, como se indica en el cuadro 8-1 y se especifica en las subcláusulas 8.2.2.4-8.2.2.6.

**Cuadro 8-1 – Tipo de mapeado de grupos de sectores refinado**

<b>slice_group_map_type</b>	<b>slice_group_change_direction_flag</b>	<b>Tipo de mapeado de grupos de sector perfeccionado</b>
3	0	Cuadrangular dextrógiro centrífugo
3	1	Cuadrangular levógiro centrífugo
4	0	Barrido por filas
4	1	Barrido inverso por filas
5	0	De izquierda a derecha
5	1	De derecha a izquierda

En ese caso, las unidades de mapeado de grupos de sectores MapUnitsInSliceGroup0 en el orden creciente especificado se asignan al grupo de sectores 0 y el resto de unidades de mapeado de grupos de sectores PicSizeInMapUnits – MapUnitsInSliceGroup0 de la imagen se asignan al grupo de sectores 1.

Cuando num\_slice\_groups\_minus1 es igual a 1 y slice\_group\_map\_type es igual a 4 ó 5, la variable sizeOfUpperLeftGroup se define como:

$$\text{sizeOfUpperLeftGroup} = (\text{slice\_group\_change\_direction\_flag} ? (\text{PicSizeInMapUnits} - \text{MapUnitsInSliceGroup0}) : \text{MapUnitsInSliceGroup0}) \quad (8-15)$$

La variable mapUnitToSliceGroupMap se calcula del modo siguiente:

- Si num\_slice\_groups\_minus1 es igual a 0, el mapeado de unidades de mapeado en grupos de sectores se genera para todos los i comprendidos entre 0 y PicSizeInMapUnits – 1, inclusive, de la siguiente manera:

$$\text{mapUnitToSliceGroupMap}[i] = 0 \quad (8-16)$$

- De lo contrario (num\_slice\_groups\_minus1 es distinto de 0), mapUnitToSliceGroupMap se calcula del modo siguiente:
  - Si slice\_group\_map\_type es igual a 0, mapUnitToSliceGroupMap se calcula según lo especificado en la subcláusula 8.2.2.1.
  - De lo contrario, si slice\_group\_map\_type es igual a 1, mapUnitToSliceGroupMap se calcula según lo especificado en la subcláusula 8.2.2.2.
  - De lo contrario, si slice\_group\_map\_type es igual a 2, mapUnitToSliceGroupMap se calcula según lo especificado en la subcláusula 8.2.2.3.
  - De lo contrario, si slice\_group\_map\_type es igual a 3, mapUnitToSliceGroupMap se calcula según lo especificado en la subcláusula 8.2.2.4.
  - De lo contrario, si slice\_group\_map\_type es igual a 4, mapUnitToSliceGroupMap se calcula según lo especificado en la subcláusula 8.2.2.5.
  - De lo contrario, si slice\_group\_map\_type es igual a 5, mapUnitToSliceGroupMap se calcula según lo especificado en la subcláusula 8.2.2.6.
  - De lo contrario (slice\_group\_map\_type es igual a 6), mapUnitToSliceGroupMap se calcula según lo especificado en la subcláusula 8.2.2.7.

Después de calcular el mapUnitToSliceGroupMap, se llama al proceso definido en la subcláusula 8.2.2.8 realizar la conversión entre los dos tipos de mapeado, es decir, convertir el mapeado de unidades de mapeado en grupos de sectores mapUnitToSliceGroupMap al mapeado de macrobloques en grupos de sectores MbToSliceGroupMap. Después de calcular el mapeado de macrobloques en grupos de sectores como se describe en la subcláusula 8.2.2.8, se define la función NextMbAddress( n ) como el valor de la variable nextMbAddress calculada del siguiente modo:

```

i = n + 1
while( i < PicSizeInMbs && MbToSliceGroupMap[ i ] != MbToSliceGroupMap[ n ] )
    i++;
nextMbAddress = i

```

(8-17)

### 8.2.2.1 Especificación del mapeado de grupos de sectores de tipo entrelazado

Las especificaciones de esta subcláusula se aplican cuando slice\_group\_map\_type es igual a 0.

El mapeado de unidades de mapeado en grupos de sectores se genera del modo siguiente:

```

i = 0
do
    for( iGroup = 0; iGroup <= num_slice_groups_minus1 && i < PicSizeInMapUnits;
        i += run_length_minus1[ iGroup++ ] + 1 )
        for( j = 0; j <= run_length_minus1[ iGroup ] && i + j < PicSizeInMapUnits; j++ )
            mapUnitToSliceGroupMap[ i + j ] = iGroup
while( i < PicSizeInMapUnits )

```

(8-18)

### 8.2.2.2 Especificación del mapeado de grupos de sectores de tipo disperso

Las especificaciones de esta subcláusula se aplican cuando slice\_group\_map\_type es igual a 1.

El mapeado de unidades de mapeado en grupos de sectores se calcula del modo siguiente:

```

for( i = 0; i < PicSizeInMapUnits; i++ )
    mapUnitToSliceGroupMap[ i ] = ( ( i % PicWidthInMbs ) +
        ( ( ( i / PicWidthInMbs ) * ( num_slice_groups_minus1 + 1 ) ) / 2 ) )
        % ( num_slice_groups_minus1 + 1 )

```

(8-19)

### 8.2.2.3 Especificación del mapeado de grupos de sectores de tipo primer plano con los sobrantes

Las especificaciones de esta subcláusula se aplican cuando slice\_group\_map\_type es igual a 2.

El mapeado de unidades de mapeado en grupos de sectores se obtiene del modo siguiente:

```

for( i = 0; i < PicSizeInMapUnits; i++ )
    mapUnitToSliceGroupMap[ i ] = num_slice_groups_minus1
for( iGroup = num_slice_groups_minus1 - 1; iGroup >= 0; iGroup-- ) {
    yTopLeft = top_left[ iGroup ] / PicWidthInMbs
    xTopLeft = top_left[ iGroup ] % PicWidthInMbs
    yBottomRight = bottom_right[ iGroup ] / PicWidthInMbs
    xBottomRight = bottom_right[ iGroup ] % PicWidthInMbs
    for( y = yTopLeft; y <= yBottomRight; y++ )
        for( x = xTopLeft; x <= xBottomRight; x++ )
            mapUnitToSliceGroupMap[ y * PicWidthInMbs + x ] = iGroup
}

```

(8-20)

Después de aplicar el proceso definido mediante la ecuación 8-20, habrá al menos un valor de  $i$  comprendido entre 0 y  $\text{PicSizeInMapUnits} - 1$ , inclusive, para el cual  $\text{mapUnitToSliceGroupMap}[ i ]$  será igual a  $iGroup$ , para cada valor de  $iGroup$  comprendido entre 0 y  $\text{num\_slice\_groups\_minus1}$ , inclusive (es decir, cada grupo de sectores contendrá al menos una unidad de mapeado de grupos de sectores).

NOTA – Los rectángulos pueden estar superpuestos. El grupo de sectores 0 contiene los macrobloques que están en el rectángulo especificado por  $\text{top\_left}[ 0 ]$  y  $\text{bottom\_right}[ 0 ]$ . Los grupos de sectores que tengan un ID de grupos de sectores mayor que 0 y menor que  $\text{num\_slice\_groups\_minus1}$  contienen los macrobloques que están en el rectángulo especificado para ese grupo de sectores y que no están en el rectángulo especificado para cualquier grupo de sectores que tenga un ID de grupo de sectores menor. El grupo de sectores cuyo ID sea igual a  $\text{num\_slice\_groups\_minus1}$  contiene los macrobloques que no están en los demás grupos de sectores.



#### 8.2.2.4 Especificación del mapeado de grupos de sectores de tipo cuadrangular dextrógiro centrífugo

Las especificaciones de esta subcláusula se aplican cuando slice\_group\_map\_type vale 3.

El mapeado de unidades de mapeado en grupos de sectores se calcula del modo siguiente:

```
for( i = 0; i < PicSizeInMapUnits; i++ )
    mapUnitToSliceGroupMap[ i ] = 1
x = ( PicWidthInMbs - slice_group_change_direction_flag ) / 2
y = ( PicHeightInMapUnits - slice_group_change_direction_flag ) / 2
( leftBound, topBound ) = ( x, y )
( rightBound, bottomBound ) = ( x, y )
( xDir, yDir ) = ( slice_group_change_direction_flag - 1, slice_group_change_direction_flag )
for( k = 0; k < MapUnitsInSliceGroup0; k += mapUnitVacant ) {
    mapUnitVacant = ( mapUnitToSliceGroupMap[ y * PicWidthInMbs + x ] == 1 )
    if( mapUnitVacant )
        mapUnitToSliceGroupMap[ y * PicWidthInMbs + x ] = 0
    if( xDir == -1 && x == leftBound ) {
        leftBound = Max( leftBound - 1, 0 )
        x = leftBound
        ( xDir, yDir ) = ( 0, 2 * slice_group_change_direction_flag - 1 )
    } else if( xDir == 1 && x == rightBound ) {
        rightBound = Min( rightBound + 1, PicWidthInMbs - 1 )
        x = rightBound
        ( xDir, yDir ) = ( 0, 1 - 2 * slice_group_change_direction_flag )
    } else if( yDir == -1 && y == topBound ) {
        topBound = Max( topBound - 1, 0 )
        y = topBound
        ( xDir, yDir ) = ( 1 - 2 * slice_group_change_direction_flag, 0 )
    } else if( yDir == 1 && y == bottomBound ) {
        bottomBound = Min( bottomBound + 1, PicHeightInMapUnits - 1 )
        y = bottomBound
        ( xDir, yDir ) = ( 2 * slice_group_change_direction_flag - 1, 0 )
    } else
        ( x, y ) = ( x + xDir, y + yDir )
}
```

#### 8.2.2.5 Especificación del mapeado de grupos de sectores de tipo barrido por filas

Las especificaciones de esta subcláusula se aplican cuando slice\_group\_map\_type es igual a 4.

El mapeado de unidades de mapeado en grupos de sectores se calcula del modo siguiente:

```
for( i = 0; i < PicSizeInMapUnits; i++ )
    if( i < sizeOfUpperLeftGroup )
        mapUnitToSliceGroupMap[ i ] = slice_group_change_direction_flag
    else
        mapUnitToSliceGroupMap[ i ] = 1 - slice_group_change_direction_flag
```

#### 8.2.2.6 Especificación del mapeado de grupos de sectores de tipo izquierda a derecha

Las especificaciones de esta subcláusula se aplican cuando slice\_group\_map\_type es igual a 5.

El mapeado de unidades de mapeado en grupos de sectores se calcula del modo siguiente:

```
k = 0;
for( j = 0; j < PicWidthInMbs; j++ )
    for( i = 0; i < PicHeightInMapUnits; i++ )
        if( k++ < sizeOfUpperLeftGroup )
            mapUnitToSliceGroupMap[ i * PicWidthInMbs + j ] = slice_group_change_direction_flag
        else
            mapUnitToSliceGroupMap[ i * PicWidthInMbs + j ] = 1 - slice_group_change_direction_flag
```

### 8.2.2.7 Especificación del mapeado de grupos de sectores de tipo explícito

Las especificaciones de esta subcláusula se aplican cuando `slice_group_map_type` es igual a 6.

El mapeado de unidades de mapeado en grupos de sectores se calcula del modo siguiente:

$$\text{mapUnitToSliceGroupMap}[ i ] = \text{slice\_group\_id}[ i ] \quad (8-24)$$

para todo  $i$  comprendido entre 0 y  $\text{PicSizeInMapUnits} - 1$ , inclusive.

### 8.2.2.8 Especificación de la conversión del mapeado de unidades de mapeado en grupos de sectores al mapeado de macrobloques en grupos de sectores

Para cada valor de  $i$  comprendido entre 0 y  $\text{PicSizeInMbs} - 1$ , inclusive, el mapeado de macrobloques en grupos de sectores se define del modo siguiente:

- Si `frame_mbs_only_flag` es igual a 1 o `field_pic_flag` es igual a 1, el mapeado de grupos de macrobloques en grupos de sectores es:

$$\text{MbToSliceGroupMap}[ i ] = \text{mapUnitToSliceGroupMap}[ i ] \quad (8-25)$$

- De lo contrario, si `MbaffFrameFlag` es igual a 1, el mapeado de macrobloques en grupos de sectores es:

$$\text{MbToSliceGroupMap}[ i ] = \text{mapUnitToSliceGroupMap}[ i / 2 ] \quad (8-26)$$

- De lo contrario (`frame_mbs_only_flag` es igual a 0 y `mb_adaptive_frame_field_flag` es igual a 0 y `field_pic_flag` es igual a 0), el mapeado de macrobloques en grupos de sectores es:

$$\text{MbToSliceGroupMap}[ i ] = \text{mapUnitToSliceGroupMap}[ ( i / ( 2 * \text{PicWidthInMbs} ) ) * \text{PicWidthInMbs} + ( i \% \text{PicWidthInMbs} ) ] \quad (8-27)$$

### 8.2.3 Proceso de decodificación de particiones de datos de sector

Este proceso acepta como argumentos:

- una RBSP de la capa partición A de datos de sector;
- si los datos de sector contienen elementos sintácticos de categoría 3, una RBSP de la capa partición B de datos de sector con el mismo `slice_id` que la RBSP de capa de partición A de datos de sectores; y
- cuando los datos de sector contienen elementos sintácticos de la categoría 4, una RBSP de capa partición C de datos de sector con idéntico `slice_id` que la RBSP de capa partición A de datos de sector.

NOTA – La RBSP de capa partición B de datos sector y la RBSP de capa de partición C de datos de sector no es necesario que estén presentes.

Este proceso genera como resultado un sector codificado.

Cuando no se utilizan particiones de datos de sectores, los sectores codificados se representan mediante una RBSP sin particiones de capa de sector que contiene la cabecera de sector seguido de la estructura sintáctica datos de sector que contienen todos los elementos sintácticos de las categorías 2, 3 y 4 (véase la columna categoría en la subcláusula 7.3) de los datos de macrobloque correspondientes a los macrobloques del sector.

Cuando se utiliza la partición de datos de sector, los datos de macrobloque de un sector se subdividen en una, dos o tres particiones contenidas en unidades NAL separadas. La partición A contiene la cabecera de la partición A de datos de sector, y todos los elementos sintácticos de la categoría 2. La partición B, si la hubiere, contiene la cabecera de la partición B de datos de sector y todos los elementos sintácticos de la categoría 3. La partición C, si la hubiere, contiene la cabecera de la partición C de datos de sector y todos los elementos sintácticos de la categoría 4.

Cuando se utilizan particiones de datos de sector, los elementos sintácticos de cada categoría se analizan sintácticamente desde una unidad NAL separada, la cual no es necesario que esté presente si no hay símbolos de la respectiva categoría. El proceso de decodificación procesará las particiones de datos de sector de un sector codificado de manera equivalente al procesamiento de una RBSP sin particiones de capa de sector, para lo cual extraerá cada elemento sintáctico de la partición de datos de sector en la que aparece dicho elemento sintáctico en función de la asignación de partición de datos de sector en los cuadros sintácticos de la subcláusula 7.3.

NOTA – Los elementos sintácticos de la categoría 3 son importantes para decodificar los datos residuales de macrobloques de tipos I y SI. Los elementos sintácticos de la categoría 4 se utilizan para decodificar los datos residuales de macrobloques de tipos P y B. La categoría 2 abarca todos los demás elementos sintácticos relacionados con la decodificación de macrobloques, y su información se indica a menudo como información de cabecera. La cabecera de la partición A de datos de sector contienen todos los elementos sintácticos de la cabecera de sector y además el `slice_id` que se utiliza para relacionar las particiones B y C de datos de sector con la partición A de datos de sector. Las cabeceras de las particiones B y C de datos de sector contienen el elemento sintáctico `slice_id` que sirve para establecer la relación de estas particiones con la partición A de datos de sector del sector.

#### **8.2.4 Proceso de decodificación para crear listas de imágenes de referencia**

Se llama a este proceso al comenzar la decodificación de cada sector P, SP o B.

Este proceso genera como resultado una lista de imágenes de referencia `RefPicList0` y, si se trata de un sector B, una segunda lista de imágenes de referencia `RefPicList1`.

Las imágenes de referencia decodificadas se marcan como "utilizada como referencia de corto alcance" o "utilizado como referencia de largo alcance" según se especifica en el tren de bits y se define en la subcláusula 8.2.5. Las imágenes de referencia decodificadas de corto alcance se identifican mediante el valor de `frame_num`. Las imágenes de referencia decodificadas de largo alcance se asignan a un índice de cuadro de largo alcance, según se especifica en el tren de bits y se define en la subcláusula 8.2.5.

Se invoca la subcláusula 8.4.2.1 para especificar:

- la asignación de las variables `FrameNum`, `FrameNumWrap`, y `PicNum` a cada una de las imágenes de referencia de corto alcance; y
- la asignación de la variable `LongTermPicNum` a cada una de las imágenes de referencia de largo alcance.

Las imágenes de referencia se direccionan mediante índice de referencia, como se especifica en la subcláusula 8.4.2.1. Un índice de referencia es un índice a una lista de variables `PicNum` y `LongTermPicNum`, la cual se denomina lista de imágenes de referencia. Al decodificar un sector P o SP, sólo se utiliza una lista de imágenes de referencia `RefPicList0`. Al decodificar un sector B, habrá una segunda lista de imágenes de referencia independiente, `RefPicList1`, además de `RefPicList0`.

Al comenzar a decodificar cada sector, la lista de imágenes de referencia `RefPicList0`, y `RefPicList1`, si se trata de un sector B, se calcula del modo siguiente:

- Se calcula una lista inicial de imágenes de referencia `RefPicList0`, y `RefPicList1` si se trata de un sector B, según se especifica en la subcláusula 8.2.4.2.
- La lista inicial de imágenes de referencia `RefPicList0` y para los sectores B `RefPicList1` se modifica como se describe en la subcláusula 8.2.4.3.

El número de elementos en la lista modificada de imágenes de referencia `RefPicList0` es `num_ref_idx_l0_active_minus1 + 1`, y si se trata de un sector B el número de elementos en la lista de imágenes de referencia modificada `RefPicList1` es `num_ref_idx_l1_active_minus1 + 1`. Una

imagen de referencia puede aparecer en más de un índice de las listas de imágenes de referencia modificadas RefPicList0 o RefPicList1.

#### 8.2.4.1 Proceso de decodificación del número de imagen

Se invoca este proceso cuando se invoca el proceso de construcción de listas de imágenes de referencia especificado en la subcláusula 8.2.4 o el proceso de marcación de imágenes de referencia especificado en la subcláusula 8.2.5.

Las variables FrameNum, FrameNumWrap, PicNum, LongTermFrameIdx, y LongTermPicNum se utilizan en el proceso de inicialización de listas de imágenes de referencia que se describe en la subcláusula 8.2.4.2, en el proceso de modificación de listas de imágenes de referencia que se describe en la subcláusula 8.2.4.3 y en el proceso de marcado de imágenes de referencia decodificadas que se describe en la subcláusula 8.2.5.

A cada imagen de referencia de corto alcance se le asignan las variables FrameNum y FrameNumWrap del modo siguiente. En primer lugar, se asigna a FrameNum el valor del elemento sintáctico frame\_num que se ha decodificado en la cabecera o cabeceras de sector de la correspondiente imagen de referencia de corto alcance. A continuación se calcula la variable FrameNumWrap:

```
if( FrameNum > frame_num )
    FrameNumWrap = FrameNum - MaxFrameNum
else
    FrameNumWrap = FrameNum
```

(8-28)

donde el valor de frame\_num utilizado en la ecuación 8-28 es el valor de frame\_num que figura en la cabecera o cabeceras de sector de la imagen considerada.

A cada imagen de referencia de largo alcance se le asigna la variable LongTermFrameIdx según se especifica en la subcláusula 8.2.5.

A cada imagen de referencia de corto alcance se le asigna una variable PicNum, y se le asigna una variable LongTermPicNum. Los valores de estas variables dependen del valor del field\_pic\_flag y del bottom\_field\_flag de la imagen considerada y se obtienen del modo siguiente:

- Si field\_pic\_flag es igual a 0, se aplica lo siguiente:
  - Para cada cuadro de referencia de corto alcance o par de campos de referencia complementarios:

$PicNum = FrameNumWrap$  (8-29)

- Para cada cuadro de referencia de largo alcance o par de campos de referencia complementarios de largo alcance:

$LongTermPicNum = LongTermFrameIdx$  (8-30)

NOTA – Al decodificar un cuadro el valor de MbaffFrameFlag no tiene efecto alguno en los cálculos definidos en las subcláusulas 8.2.4.2, 8.2.4.3 y 8.2.5.

- De lo contrario (field\_pic\_flag es igual a 1), se aplica lo siguiente:
  - Para cada campo de referencia de corto alcance se aplica lo siguiente:
    - Si el campo de referencia tiene la misma paridad que el campo considerado

$PicNum = 2 * FrameNumWrap + 1$  (8-31)

- De lo contrario (el campo de referencia tiene paridad contraria al campo considerado):

$PicNum = 2 * FrameNumWrap$  (8-32)

- Para cada campo de referencia de largo alcance se aplica lo siguiente:
  - Si el campo de referencia tiene la misma paridad que el campo considerado

$$\text{LongTermPicNum} = 2 * \text{LongTermFrameIdx} + 1 \quad (8-33)$$

- De lo contrario (el campo de referencia tiene paridad contraria al campo considerado),

$$\text{LongTermPicNum} = 2 * \text{LongTermFrameIdx} \quad (8-34)$$

### 8.2.4.2 Proceso de inicialización de listas de imágenes de referencia

Se llama este proceso de inicialización al decodificar cabeceras de sector P, SP, o B.

Este proceso genera como resultado la lista inicial de imágenes de referencia RefPicList0, y si se trata de un sector B, la lista inicial de imágenes de referencia RefPicList1.

RefPicList0 y RefPicList1 contienen elementos iniciales de las variables PicNum y LongTermPicNum, como se describe en las subcláusulas 8.2.4.2.1 a 8.2.4.2.5.

Cuando el número de elementos en la RefPicList0 o RefPicList1 inicial, generado como se especifica en las subcláusulas 8.2.4.2.1 a 8.2.4.2.5, es mayor que respectivamente, num\_ref\_idx\_l0\_active\_minus1 + 1 o num\_ref\_idx\_l1\_active\_minus1 + 1, se descartan los elementos cuya posición es posterior a num\_ref\_idx\_l0\_active\_minus1 o num\_ref\_idx\_l1\_active\_minus1 en la lista de imágenes de referencia inicial.

Cuando el número de elementos de la RefPicList0 o RefPicList1, generado según se describe en las subcláusulas 8.2.4.2.1 a 8.2.4.2.5, es inferior a num\_ref\_idx\_l0\_active\_minus1 + 1 o num\_ref\_idx\_l1\_active\_minus1 + 1, respectivamente, los elementos restantes en la lista de imágenes de referencia inicial se les asigna el valor de "sin imagen de referencia".

#### 8.2.4.2.1 Proceso de inicialización de listas de imágenes de referencia para sectores P y SP en cuadros

Se llama a este proceso de inicialización al decodificar sectores P o SP en un cuadro codificado.

Este proceso genera como resultado la lista de imágenes de referencia inicial RefPicList0.

Cuando se invoca este proceso, habrá por lo menos un cuadro de referencia o un par de campos de referencia complementarios que se encuentre marcado como "utilizado para referencia de corto alcance" o "utilizado para referencia de largo alcance".

El orden de la lista de imágenes de referencia RefPicList0 es tal que los cuadros de referencia de corto alcance y los pares de campos de referencia complementarios de corto alcance tienen un índice menor que los cuadros de referencia de largo alcance y los pares de campos de referencia complementarios de largo alcance.

Los cuadros de referencia y los pares de campo de referencia complementarios de corto alcance se ordenan comenzando con el cuadro de referencia o el par de campos de referencia complementarios de largo alcance de PicNum y continuando en orden descendente hasta el cuadro de referencia o el par de campos de referencia complementarios con el valor más bajo de PicNum.

Los cuadros de referencia y los pares de campo de referencia complementarios de largo alcance se ordenan comenzando con el cuadro de referencia o el par de campos de referencia complementarios de corto alcance de LongTermPicNum y continuando en orden ascendente hasta el cuadro de referencia o el par de campos de referencia complementarios con el valor más alto de LongTermPicNum.

NOTA – Los campos de referencia desapareados no se utilizan en la predicción inter para decodificar un cuadro, independientemente del valor de MbaffFrameFlag.

Por ejemplo, si hay tres cuadros de referencia marcados como "utilizados para referencia de corto alcance" con PicNum igual a 300, 302 y 303 y dos cuadros de referencia marcados como "utilizados para referencia de largo alcance" con LongTermPicNum igual a 0 y 3 el orden inicial será:

- a RefPicList0[0] se le da el valor de la imagen de referencia de corto alcance con PicNum = 303;

- a RefPicList0[1] se le da el valor de la imagen de referencia de corto alcance con PicNum = 302;
- a RefPicList0[2] se le da el valor de la imagen de referencia de corto alcance con PicNum = 300;
- a RefPicList0[3] se le da el valor de la imagen de referencia de corto alcance con LongTermPicNum = 0; y
- a RefPicList0[4] se le da el valor de la imagen de referencia de corto alcance con LongTermPicNum = 3.

#### **8.2.4.2.2 Proceso de inicialización de listas de imágenes de referencia para sectores P y SP en campos**

Se llama a este proceso inicialización al decodificar un sector P o SP en un campo codificado.

Este proceso genera como resultado la lista inicial de imágenes de referencia RefPicList0.

Al decodificar un campo, cada campo incluido en la lista de imágenes de referencia tiene un índice distinto en la lista.

NOTA – Al decodificar un campo, en realidad habrá como mínimo el doble de imágenes disponibles para referencia de las que habría al decodificar un cuadro en la misma posición en orden de decodificación.

Se calculan las dos listas ordenadas de cuadros de referencia, refFrameList0ShortTerm y refFrameList0LongTerm, del modo siguiente. Para la creación de esta lista de cuadros, se consideran cuadros de referencia los cuadros decodificados, los pares de campos de referencia complementarios, los campos de referencia desapareados y los cuadros de referencia en los cuales sólo hay un campo marcado como "utilizado para referencia de corto alcance" o "utilizado para referencia de largo alcance".

- En la lista de cuadros de referencia de corto alcance refFrameList0ShortTerm se incluye el FrameNumWrap de todos los cuadros que tengan algún campo marcado como "utilizado para referencia de corto alcance". Si el campo considerado es el segundo campo (en orden de decodificación) de un par de campos de referencia complementarios y el primer campo está marcado como "utilizado para referencia de corto alcance", el FrameNumWrap del primer campo se incluye en la lista refFrameList0ShortTerm. Los cuadros de la lista refFrameList0ShortTerm se ordena en orden decreciente de FrameNumWrap.
- En la lista de cuadros de referencia de largo alcance refFrameList0LongTerm se incluye el LongTermFrameIdx de todos los cuadros que tengan algún campo marcado como "utilizado como referencia de largo alcance". Cuando el campo considerado es el segundo campo (en orden de decodificación) de un par de campos de referencia complementarios y el primer campo está marcado como "utilizado como referencia de largo alcance", el LongTermFrameIdx del primer campo se incluye en la lista de refFrameList0LongTerm. Los cuadros en esta lista refFrameList0LongTerm se ordena en orden creciente de LongTermFrameIdx.

Se llama al proceso descrito en la subcláusula 8.2.4.2.5, pasándole como argumentos refFrameList0ShortTerm y refFrameList0LongTerm, y el resultado se asigna a RefPicList0.

#### **8.2.4.2.3 Proceso de inicialización de listas de imágenes de referencia para sectores B en cuadros**

Se llama este proceso de inicialización al decodificar un sector B en un cuadro codificado.

Este proceso genera como resultado las listas de imágenes de referencia iniciales RefPicList0 y RefPicList1.

Cuando se invoca este proceso, habrá por lo menos un cuadro de referencia o un par de campos de referencia complementarios que se encuentre marcado como "utilizado para referencia de corto alcance" o "utilizado para referencia de largo alcance".

Para los sectores B, el orden de las imágenes de referencia de corto alcance en las listas de imágenes de referencia RefPicList0 y RefPicList1 depende del orden de salida, generado por PicOrderCnt( ). Si pic\_order\_cnt\_type es igual a 0, no se incluyen en RefPicList0 ni en RefPicList1 las imágenes de referencia que están marcadas como "inexistente" según se describe en la subcláusula 8.2.5.2.

NOTA – Si gaps\_in\_frame\_num\_value\_allowed\_flag es igual a 1, los codificadores deben garantizar el correcto funcionamiento del proceso de decodificación mediante el empleo de reordenamiento de lista de imágenes de referencia (en particular cuando pic\_order\_cnt\_type es igual a 0, en cuyo caso no se infiere PicOrderCnt( ) para el caso de cuadros "inexistentes").

El orden de la lista de imágenes de referencia RefPicList0 es tal que los cuadros de referencia de corto alcance y los pares de campos de referencia complementarios de corto alcance tienen índice menor que los cuadros de referencia de largo alcance y los pares de campos de referencia complementarios de largo alcance. El orden es el siguiente:

- Los cuadros de referencia de corto alcance y los pares de campos de referencia complementarios de corto alcance se ordenan de modo que el primer cuadro de referencia o par de campos de referencia complementarios frm0 sea el que tiene el valor más grande de PicOrderCnt( frm0 ) y que a la vez sea menor que el valor de PicOrderCnt( CurrPic ); después en orden descendente hasta el cuadro de referencia o par de campos de referencia complementarios de corto alcance frm1 que tiene el valor más pequeño de PicOrderCnt( frm1 ); a continuación el cuadro de referencia o par de campos de referencia complementarios de corto alcance frm2 cuyo valor sea el valor más pequeño de PicOrderCnt( frm2 ) y que a la vez sea mayor que el valor de PicOrderCnt( CurrPic ) del cuadro considerado; y después en orden ascendente hasta el cuadro de referencia o par de campos de referencia complementarios de corto alcance frm3 que tenga el valor más grande de PicOrderCnt( frm3 ).
- Los cuadros de referencia de largo alcance y los pares de campos de referencia complementarios de largo alcance se ordenan comenzando con el cuadro de referencia o el par de campos de referencia complementarios de largo alcance que tengan el valor más pequeño de LongTermPicNum y continuando en orden ascendente hasta el cuadro de referencia o el par de campos de referencia complementarios de largo alcance que tengan el valor más grande de LongTermPicNum.

La lista de imagen de referencia RefPicList1 se ordena de modo que los cuadros de referencia de corto alcance y los pares de campos de referencia complementarios de corto alcance tengan índice menor que los cuadros de referencia de largo alcance y los pares de campos de referencia complementarios de largo alcance. El orden es el siguiente:

- Los cuadros de referencia de corto alcance y los pares de campos de referencia complementarios de corto alcance se ordenan de modo que el primer cuadro de referencia o par de campos de referencia complementarios frm4 sea el que tiene el valor más pequeño de PicOrderCnt( frm4 ) y que a la vez sea mayor que el valor de PicOrderCnt( CurrPic ); y después en orden descendente hasta el cuadro de referencia o par de campos de referencia complementarios de corto alcance frm5 que tiene el valor más pequeño de PicOrderCnt( frm5 ), a continuación el cuadro de referencia o par de campos de referencia complementarios de corto alcance frm6 cuyo valor sea el valor más grande de PicOrderCnt( frm6 ) y que a la vez sea menor que el valor de PicOrderCnt( CurrPic ) del cuadro considerado; y después en orden ascendente hasta el cuadro de referencia o par de campos de referencia complementarios de corto alcance frm7 que tenga el valor más pequeño de PicOrderCnt( frm7 ).

- Los cuadros de referencia de largo alcance y los pares de campos de referencia complementarios de largo alcance se ordenan comenzando con el cuadro de referencia o el par de campos de referencia complementarios de largo alcance que tengan el valor más pequeño de LongTermPicNum y continuando en orden ascendente hasta el cuadro de referencia o el par de campos de referencia complementarios de largo alcance que tengan el valor más grande de LongTermPicNum.
  - Cuando la lista de imágenes de referencia RefPicList1 contiene más de un elemento y esta lista RefPicList1 es idéntica a la lista de imágenes de referencia RefPicList0, se intercambian los primeros dos elementos RefPicList1[0] y RefPicList1[1].
- NOTA – Los campos de referencia desapareados no se utilizan en la predicción inter de cuadros, independientemente del valor de MbaffFrameFlag.

#### **8.2.4.2.4 Proceso de inicialización de listas de imágenes de referencia para sectores B en campos**

Se llama a este proceso de inicialización al decodificar sectores B en un campo codificado.

Este proceso genera como resultado las listas de imágenes de referencia iniciales RefPicList0 y RefPicList1.

Al decodificar un campo, cada campo de un cuadro de referencia almacenado se identifica como una imagen de referencia distinta con un índice único. El orden de las imágenes de referencia de corto alcance en las listas de imágenes de referencia RefPicList0 y RefPicList1 depende del orden de salida, generado por PicOrderCnt( ). Si pic\_order\_cnt\_type es igual a 0, no se incluyen en RefPicList0 ni en RefPicList1 las imágenes de referencia que están marcadas como "inexistente" según se describe en la subcláusula 8.2.5.2.

NOTA – Si gaps\_in\_frame\_num\_value\_allowed\_flag es igual a 1, los codificadores deben garantizar el correcto funcionamiento del proceso de decodificación mediante el empleo de reordenamiento de lista de imágenes de referencia (en particular cuando pic\_order\_cnt\_type es igual a 0, en cuyo caso no se infiere PicOrderCnt( ) para el caso de cuadros "inexistentes").

NOTA – Al decodificar un campo, habrá en realidad al menos dos veces el número de imágenes disponibles para referencia del que habría al decodificar un cuadro en la misma posición en orden de decodificación.

Las tres listas ordenadas de cuadro de referencia refFrameList0ShortTerm, refFrameList1ShortTerm y refFrameListLongTerm se calculan del modo siguiente. A efectos de la creación de estas listas de cuadro, se utiliza el término elemento de referencia para designar a cuadros de referencia decodificados, pares de campos de referencia complementarios o campos de referencia desapareados. Si pic\_order\_cnt\_type es igual a 0, el término entrada de referencia no hace alusión a cuadros que están marcados como "inexistentes" según lo descrito en la subcláusula 8.2.5.2.

- refFrameList0ShortTerm se ordena de modo que el primer elemento de referencia f0 sea el que tiene el valor más grande de PicOrderCnt( f0 ) y que a la vez es menor o igual al valor de PicOrderCnt( CurrPic ) del campo considerado; y después en orden descendente hasta el elemento de referencia de corto alcance f1 que tiene el valor más pequeño que PicOrderCnt( f1 ); a continuación el elemento de referencia f2 cuyo valor sea más pequeño de PicOrderCnt( f2 ) y que a la vez sea mayor que el valor de PicOrderCnt( CurrPic ) del campo considerado y después en orden ascendente hasta el elemento de referencia de corto alcance f3 que tiene el valor más grande de PicOrderCnt( f3 ).

NOTA – Cuando el campo considerado sigue en orden de decodificación a un campo codificado fldPrev con el que forma un par de campos de referencia complementarios, se incluye el fldPrev en la lista refFrameList0ShortTerm, para lo cual se utiliza PicOrderCnt( fldPrev ) y se aplica el método de ordenación descrito en la frase anterior.



- refFrameList1ShortTerm se ordena de modo que el primer elemento de referencia f4 sea el que tiene el valor más pequeño de PicOrderCnt( f4 ) y que a la vez sea mayor que el valor de PicOrderCnt( CurrPic ) del campo considerado; a continuación en orden ascendente hasta el elemento de referencia de corto alcance f5 que tiene el valor más grande de PicOrderCnt( f5 ); a continuación con el elemento de referencia f6 cuyo valor sea más grande de PicOrderCnt( f6 ) y a la vez sea menor o igual que el valor de PicOrderCnt( CurrPic ) del campo considerado; y después en orden descendente hasta el elemento de referencia de corto alcance f7 que tiene el valor más pequeño de PicOrderCnt( f7 ).

NOTA – Cuando el campo considerado aparece después en el orden de decodificación de un campo codificado fldPrev con el que forma un par de campos de referencia complementarios, se incluye el fldPrev en la lista refFrameList1ShortTerm utilizando PicOrderCnt( fldPrev ) y se aplica el método de ordenación descrito en la frase anterior.

- refFrameListLongTerm se ordena de modo que el primer elemento de referencia sea el que tiene el valor más pequeño de LongTermFrameIdx y después en orden ascendente hasta el elemento de referencia con el mayor valor de LongTermFrameIdx.

NOTA – Cuando el campo complementario de la imagen considerada está marcada como "utilizado como referencia de largo alcance", se incluye en la lista refFrameListLongTerm. Los elementos de referencia que sólo tienen un campo marcado como "utilizado como referencia a largo alcance" se incluyen en la lista refFrameListLongTerm.

Se llama al proceso definido en la subcláusula 8.2.4.2.5, pasándolo como argumentos refFrameList0ShortTerm y refFrameListLongTerm, y el resultado se asigna a RefPicList0.

Se llama al proceso definido en la subcláusula 8.2.4.2.5, pasándolo como argumentos refFrameList1ShortTerm y refFrameListLongTerm, y el resultado se asigna a RefPicList1.

Cuando la lista de imágenes de referencia RefPicList1 contiene más de un elemento y es idéntica a la lista de imágenes de referencia RefPicList0, se intercambian los dos primeros elementos RefPicList1[0] y RefPicList1[1].

#### **8.2.4.2.5 Proceso de inicialización de listas de imágenes de referencia en campos**

Este proceso acepta como argumentos las listas de cuadros de referencia refFrameListXShortTerm (siendo X 0 ó 1) y refFrameListLongTerm.

Este proceso genera como resultado la lista de imágenes como referencia RefPicListX (es decir RefPicList0 o RefPicList1).

La lista de imágenes de referencia RefPicListX es una lista ordenada de modo que los campos de referencia de corto alcance tienen índice menor que los campos de referencia de largo alcance. El cálculo de esta lista a partir de las listas refFrameListXShortTerm y refFrameListLongTerm, es el siguiente.

- Para ordenar los campos de referencia de corto alcance se seleccionan los campos de referencia de la lista ordenada de cuadros refFrameListXShortTerm alternando entre los campos de distinta paridad, y empezando por el campo que tiene la misma paridad que el campo considerado (si está presente). Cuando un campo de un cuadro de referencia no se ha decodificado o no está marcado como "utilizado como referencia de corto alcance" no se tiene en cuenta el campo que falta y en su lugar se incluye en RefPicListX el siguiente campo de referencia almacenado disponible con la paridad elegida de la lista ordenada de cuadros refFrameListXShortTerm. Cuando no haya más campos de referencia de corto alcance de paridad alternada en la lista ordenada de cuadros refFrameListXShortTerm, se insertarán en RefPicListX los siguientes campos indexados que aún no están ordenados, la paridad disponible en el orden en que aparecen en la lista ordenada de cuadros refFrameListXShortTerm.

- Para ordenar los campos de referencia de largo alcance se seleccionan los campos de referencia de la lista ordenada de cuadros refFrameListLongTerm alternando entre los campos de distinta paridad, y empezando por un campo que tiene la misma paridad que el campo considerado (si está presente). Cuando un campo de un cuadro de referencia no se ha decodificado o no está marcado como "utilizado como referencia de largo alcance" no se tiene en cuenta el campo que falta y en su lugar se incluye en RefPicListX el siguiente campo de referencia almacenado disponible con la paridad elegida de la lista ordenada de cuadros refFrameListLongTerm. Cuando no haya más campos de referencia de largo alcance de paridad alternada en la lista ordenada de cuadros refFrameListLongTerm, se insertarán en RefPicListX los siguientes campos indexados que aún no están ordenados de la paridad disponible en el orden en que aparecen en la lista ordenada de cuadros refFrameListLongTerm.

### 8.2.4.3 Proceso de reordenación de listas de imágenes de referencia

Este proceso acepta como argumento la lista de imágenes de referencia RefPicList0, y en la decodificación de sectores B, acepta además la lista de imágenes de referencia RefPicList1.

Este proceso genera como resultado una posible lista modificada de imágenes de referencia RefPicList0 y, en la decodificación de sectores B también una posible lista modificada de imágenes de referencia RefPicList1.

Cuando ref\_pic\_list\_reordering\_flag\_l0 es igual a 1, se aplica lo siguiente.

- Sea refIdxL0 un índice en la lista de imágenes de referencia RefPicList0, cuyo valor inicial es 0.
- Los correspondientes elementos sintácticos reordering\_of\_pic\_nums\_idc se procesan en el orden que aparecen en el tren de bits. Para cada uno de estos elementos sintácticos se aplica lo siguiente.
  - Si reordering\_of\_pic\_nums\_idc es igual a 0 ó 1, se llama al proceso especificado en la subcláusula 8.2.4.3.1 pasándole como argumentos RefPicList0 y refIdxL0, y el resultado se asigna a RefPicList0 y refIdxL0.
  - De lo contrario, si reordering\_of\_pic\_nums\_idc es igual a 2, se llama al proceso especificado en la subcláusula 8.2.4.3.2 pasándole como argumentos RefPicList0 y refIdxL0 y el resultado se asigna a RefPicList0 y refIdxL0.
  - De lo contrario (reordering\_of\_pic\_nums\_idc es igual a 3), se termina el proceso de reordenación de lista de imágenes de referencia RefPicList0.

Cuando ref\_pic\_list\_reordering\_flag\_l1 es igual a 1, se aplica lo siguiente.

- Sea refIdxL1 un índice en la lista de imágenes de referencia RefPicList1, cuyo valor inicial es igual a 0.
- Los correspondientes elementos sintácticos reordering\_of\_pic\_nums\_idc se procesan en el orden que aparecen en el tren de bits. Para cada uno de estos elementos sintácticos se aplica lo siguiente.
  - Si reordering\_of\_pic\_nums\_idc es igual a 0 ó 1, se llama al proceso especificado en la subcláusula 8.2.4.3.1 pasándole como argumentos RefPicList1 y refIdxL1, y el resultado se asigna a RefPicList1 y refIdxL1.
  - De lo contrario, si reordering\_of\_pic\_nums\_idc es igual a 2, se llama al proceso especificado en la subcláusula 8.2.4.3.2 pasándole como argumentos RefPicList1 y refIdxL1 y el resultado se asigna a RefPicList1 y refIdxL1.
  - De lo contrario (reordering\_of\_pic\_nums\_idc es igual a 3), se termina el proceso de reordenación de lista de imágenes de referencia RefPicList1.

### 8.2.4.3.1 Proceso de reordenación de listas de imágenes de referencia para imágenes de corto alcance

Este proceso acepta como argumentos la lista de imágenes de referencia RefPicListX (siendo X 0 ó 1) y un índice refldxLX de esta lista.

Este proceso genera como resultado una posible lista modificada de imágenes de referencia RefPicListX y el índice refldxLX incrementado.

La variable picNumLXNoWrap se calcula del modo siguiente.

- Si reordering\_of\_pic\_nums\_idc es igual a 0

$$\begin{aligned} & \text{if( picNumLXPred - ( abs\_diff\_pic\_num\_minus1 + 1 ) < 0 )} \\ & \quad \text{picNumLXNoWrap} = \text{picNumLXPred} - ( \text{abs\_diff\_pic\_num\_minus1} + 1 ) + \text{MaxPicNum} \quad (8-35) \\ & \text{else} \\ & \quad \text{picNumLXNoWrap} = \text{picNumLXPred} - ( \text{abs\_diff\_pic\_num\_minus1} + 1 ) \end{aligned}$$

- De lo contrario (reordering\_of\_pic\_nums\_idc es igual a 1),

$$\begin{aligned} & \text{if( picNumLXPred + ( abs\_diff\_pic\_num\_minus1 + 1 ) >= \text{MaxPicNum} )} \\ & \quad \text{picNumLXNoWrap} = \text{picNumLXPred} + ( \text{abs\_diff\_pic\_num\_minus1} + 1 ) - \text{MaxPicNum} \quad (8-36) \\ & \text{else} \\ & \quad \text{picNumLXNoWrap} = \text{picNumLXPred} + ( \text{abs\_diff\_pic\_num\_minus1} + 1 ) \end{aligned}$$

picNumLXPred es el valor de predicción de la variable picNumLXNoWrap. Cuando se llama por primera vez al proceso definido en esta subcláusula para un sector (es decir, para la primera vez de reordering\_of\_pic\_nums\_idc igual a 0 ó 1 en la sintaxis ref\_pic\_list\_reordering()), picNumL0Pred y picNumL1Pred se ponen inicialmente a CurrPicNum. Después de cada asignación de picNumLXNoWrap, el valor de picNumLXNoWrap se asigna a picNumLXPred.

La variable picNumLX se calcula del modo siguiente

$$\begin{aligned} & \text{if( picNumLXNoWrap > CurrPicNum )} \\ & \quad \text{picNumLX} = \text{picNumLXNoWrap} - \text{MaxPicNum} \quad (8-37) \\ & \text{else} \\ & \quad \text{picNumLX} = \text{picNumLXNoWrap} \end{aligned}$$

picNumLX será igual al PicNum de una imagen de referencia que está marcada como "utilizada para referencia de corto alcance" y no será igual al PicNum de una imagen de referencia de corto alcance marcada como "inexistente".

Se utilizará el siguiente procedimiento para ubicar la imagen con el número de imágenes de corto alcance picNumLX en la posición refldxLX, desplazar hacia el final la posición del resto de imágenes de referencia en la lista e incrementar el valor de refldxLX.

$$\begin{aligned} & \text{for( cIdx = num\_ref\_idx\_lX\_active\_minus1 + 1; cIdx > refldxLX; cIdx-- )} \\ & \quad \text{RefPicListX[ cIdx ]} = \text{RefPicListX[ cIdx - 1 ]} \\ & \text{RefPicListX[ refldxLX++ ]} = \text{imagen de referencia de corto alcance con PicNum igual a picNumLX} \\ & \text{nIdx} = \text{refldxLX} \\ & \text{for( cIdx = refldxLX; cIdx <= num\_ref\_idx\_lX\_active\_minus1 + 1; cIdx++ )} \quad (8-38) \\ & \quad \text{if( PicNumF( RefPicListX[ cIdx ] ) != picNumLX )} \\ & \quad \quad \text{RefPicListX[ nIdx++ ]} = \text{RefPicListX[ cIdx ]} \end{aligned}$$

donde la función PicNumF( RefPicListX[ cIdx ] ) se calcula de la siguiente manera:

- Si la imagen RefPicListX[ cIdx ] está marcada como "utilizada para referencia de corto alcance", entonces PicNumF( RefPicListX[ cIdx ] ) es el PicNum de la imagen RefPicListX[ cIdx ].
- De lo contrario (la imagen RefPicListX[ cIdx ] no está marcada como "utilizada para referencia de corto alcance") entonces PicNumF( RefPicListX[ cIdx ] ) es igual a MaxPicNum.

NOTA – MaxPicNum nunca puede tener un valor igual a picNumLX.

NOTA – En este procedimiento de pseudocódigo, la longitud de la lista RefPicListX aumenta temporalmente en un elemento con respecto a la longitud necesaria para la lista final. Después de ejecutar este procedimiento sólo permanecen los elementos entre 0 y num\_ref\_idx\_IX\_active\_minus1.

#### 8.2.4.3.2 Proceso de reordenación de listas de imágenes de referencia para imágenes de largo alcance

Este proceso acepta como argumentos la lista de imágenes de referencia RefPicListX (siendo X 0 ó 1) y un índice refIdxLX de esta lista.

Este proceso genera como resultado una posible lista de imágenes de referencia modificada RefPicListX y el índice refIdxLX incrementado.

Se utilizará el siguiente procedimiento para ubicar la imagen con número de imagen de largo alcance long\_term\_pic\_num en la posición refIdxLX, desplazar hacia el final la posición de las demás imágenes de la lista e incrementar el valor de refIdxLX.

```

for( cIdx = num_ref_idx_IX_active_minus1 + 1; cIdx > refIdxLX; cIdx-- )
    RefPicListX[ cIdx ] = RefPicListX[ cIdx - 1 ]
RefPicListX[ refIdxLX++ ] = imagen de referencia de largo alcance con LongTermPicNum igual a
long_term_pic_num
nIdx = refIdxLX
for( cIdx = refIdxLX; cIdx <= num_ref_idx_IX_active_minus1 + 1; cIdx++ )
    if( LongTermPicNumF( RefPicListX[ cIdx ] ) != long_term_pic_num )
        RefPicListX[ nIdx++ ] = RefPicListX[ cIdx ]

```

(8-39)

donde la función LongTermPicNumF( RefPicListX[ cIdx ] ) se calcula de la siguiente manera:

- Si la imagen RefPicListX[ cIdx ] está marcada como "utilizada para referencia de largo alcance", LongTermPicNumF( RefPicListX[ cIdx ] ) es igual a LongTermPicNum de la imagen RefPicListX[ cIdx ].
- De lo contrario (la imagen RefPicListX[ cIdx ] no está marcada como "utilizada para referencia de largo alcance"), LongTermPicNumF( RefPicListX[ cIdx ] ) es igual a  $2 * ( \text{MaxLongTermFrameIdx} + 1 )$ .

NOTA – El valor de  $2 * ( \text{MaxLongTermFrameIdx} + 1 )$  nunca puede ser igual a long\_term\_pic\_num.

NOTA – En este procedimiento de pseudocódigo la longitud de la lista RefPicListX aumenta temporalmente en un elemento con respecto a la longitud final de la lista. Después de ejecutar este procedimiento sólo se retienen los elementos comprendidos entre 0 y num\_ref\_idx\_IX\_active\_minus1.

#### 8.2.5 Proceso de marcado de imágenes de referencia decodificadas

Se llama a este proceso cuando nal\_ref\_idc es distinto de 0.

NOTA – Un proceso (el proceso de decodificación de vacíos en frame\_num especificado en la subcláusula 8.2.5.2) definido en esta subcláusula también se puede invocar cuando nal\_ref\_idc es igual a 0, como se especifica en la cláusula 8.

Las imágenes decodificadas con nal\_ref\_idc distinto de 0, denominadas imágenes de referencia, están marcadas como "utilizadas para referencia de corto alcance" o "utilizadas para referencia de largo alcance". Los dos campos de un cuadro de referencia decodificado están marcados de igual manera que el cuadro. El par de campos de referencia complementarios están marcados de igual manera que sus dos campos. Las imágenes marcadas como "utilizada como referencia de corto alcance" se identifican mediante su FrameNum y, si se trata de un campo, por su paridad. Las imágenes marcadas, "utilizada como referencia de largo alcance" se identifica mediante su LongTermFrameIdx y, si se trata de un campo, por su paridad.

Los cuadros o pares de campos complementarios marcados como "utilizado como referencia de corto alcance" o como "utilizado como referencia de largo alcance" se pueden utilizar como

referencia para la predicción inter al decodificar un cuadro hasta que el cuadro, el par de campos complementarios o uno de sus campos que lo constituyen esté marcado como "no utilizado como referencia". Los campos marcados como "utilizados para referencia de corto alcance" o como "utilizados para referencia de largo alcance" se pueden utilizar como referencia para la predicción Inter cuando se decodifica un campo hasta que esté marcado como "no utilizado como referencia".

Una imagen se puede marcar como "no utilizada como referencia" mediante el proceso de marcado de imágenes de referencia de ventana deslizante, un mecanismo de tipo primero en entrar primero en salir especificado en la subcláusula 8.2.5.3, o mediante el proceso de marcado de imágenes de referencia de control adaptativo de memoria, una operación de marcado adaptativa personalizada especificada en la subcláusula 8.2.5.4.

En el proceso de decodificación, las imágenes de referencia de corto alcance se identifican mediante sus variables FrameNum y FrameNumWrap y su número de imagen PicNum, y las imágenes de referencia de largo alcance se identifican mediante su número de imagen de largo alcance LongTermPicNum. Cuando la imagen actual no es una imagen IDR, se invoca la subcláusula 8.2.4.1 para especificar la asignación de las variables FrameNum, FrameNumWrap, PicNum y LongTermPicNum.

### **8.2.5.1 Secuencia de operaciones del proceso de marcado de imágenes de referencia decodificadas**

Los pasos necesarios para marcar imágenes de referencia decodificadas son los siguientes:

1. Cuando frame\_num de la imagen considerada es distinto de PrevRefFrameNum y es distinto de  $(\text{PrevRefFrameNum} + 1) \% \text{MaxFrameNum}$ , se aplica el proceso de decodificación de vacíos en frame\_num se realiza de acuerdo con la subcláusula 8.2.5.2.
2. Se decodifican todos los sectores de la imagen considerada.
3. En función de si la imagen considerada es una imagen IDR, se aplica lo siguiente:
  - Si la imagen considerada es una imagen IDR, se aplica lo siguiente:
    - todas las imágenes de referencia se marcarán como "no utilizadas como referencia";
    - en función de long\_term\_reference\_flag, se aplica lo siguiente:
      - si long\_term\_reference\_flag es igual a 0, la imagen IDR se marca como "utilizada como referencia de corto alcance" y MaxLongTermFrameIdx se pone a "no hay índices de cuadro de largo alcance";
      - de lo contrario (long\_term\_reference\_flag es igual a 1), la imagen IDR se marca como "utilizada como referencia de largo alcance", LongTermFrameIdx se pone a 0 y MaxLongTermFrameIdx se pone a 0.
  - De lo contrario (la imagen considerada no es una imagen IDR), se aplica lo siguiente:
    - si adaptive\_ref\_pic\_marking\_mode\_flag es igual a 0, se llama al proceso descrito en la subcláusula 8.2.5.3;
    - de lo contrario (adaptive\_ref\_pic\_marking\_mode\_flag es igual a 1), se llama al proceso descrito en la subcláusula 8.2.5.4.
4. Si la imagen considerada no es una imagen IDR y no fue marcada como "utilizada como referencia de largo alcance" mediante memory\_management\_control\_operation de valor igual a 6, la imagen se marcará como "utilizada como referencia de corto alcance".

Después de marcar la imagen de referencia decodificada considerada, el número total de cuadros con al menos un campo marcado como "utilizado como referencia" más el número de pares complementarios con al menos un campo marcado como "utilizado como referencia", más el

número de campos desapareados marcados como "utilizados como referencia", no será mayor que  $\text{Max}(\text{num\_ref\_frames}, 1)$ .

### 8.2.5.2 Proceso de decodificación de vacíos en frame\_num

Se llama a este proceso cuando frame\_num es distinto de PrevRefFrameNum y es distinto de  $(\text{PrevRefFrameNum} + 1) \% \text{MaxFrameNum}$ .

NOTA – Aunque este proceso se especifica como una subcláusula de la subcláusula 8.2.5 (que define un proceso que se invoca solamente cuando nal\_ref\_idc es diferente de 0), también puede invocarse cuando nal\_ref\_idc es igual a 0 (como se describe en la cláusula 8). Las razones por las que se ubica esta subcláusula en este sitio dentro de la estructura de esta Recomendación | Norma Internacional son históricas.

NOTA – Sólo se puede llamar a este proceso para un tren de bits conforme cuando gaps\_in\_frame\_num\_value\_allowed\_flag es igual a 1. Cuando gaps\_frame\_num\_value\_allowed\_flag es igual a 0 y frame\_num sea distinto de PrevRefFrameNum y distinto de  $(\text{PrevRefFrameNum} + 1) \% \text{MaxFrameNum}$ , el proceso de decodificación debería deducir una pérdida no intencionada de imágenes.

Cuando se llama a este proceso, se obtiene un conjunto de valores de frame\_num correspondientes a imágenes "inexistentes" que será igual a todos los valores tomados por UnusedShortTermFrameNum en la ecuación 7-10, excepto el valor de frame\_num de la imagen considerada.

El proceso de decodificación generará y marcará un cuadro para cada valor de frame\_num que corresponda a imágenes "inexistente", para lo cual se generarán los valores de UnusedShortTermFrameNum mediante la ecuación 7-10, y se utilizará el proceso de marcado de imágenes de "ventana deslizante" descrito en la subcláusula 8.2.5.3. Los cuadros generados también se marcarán como "inexistente" y "utilizados para referencia de corto alcance". Los valores de muestras de los cuadros generados pueden fijarse a cualquier valor. A estos cuadros generados que están marcados como "inexistente" no se les hará referencia en el proceso de predicción inter, no se les hará referencia en las instrucciones de reordenación de listas de imágenes de referencia para imágenes de corto alcance (subcláusula 8.2.4.3.1) y no se les hará referencia en el proceso de asignación de LongTermFrameIdx a una imagen de corto alcance (subcláusula 8.2.5.4.3).

Si pic\_order\_cnt\_type no es igual a 0, se invoca el proceso de decodificación para el número de orden de imágenes de la subcláusula 8.2.1 con el fin de calcular TopFieldOrderCnt y BottomFieldOrderCnt para cada uno de los cuadros "inexistentes". Cuando se invoca el proceso de la subcláusula 8.2.1 para un cuadro "inexistente" particular, se considera que la imagen actual es una imagen en la que se calculó que frame\_num es igual a UnusedShortTermFrameNum, que nal\_ref\_idc es diferente de 0, que nal\_unit\_type es diferente de 5, que field\_pic\_flag es igual a 0, que adaptive\_ref\_pic\_marking\_mode\_flag es igual a 0, que delta\_pic\_order\_cnt[ 0 ] (si se necesita) es igual a 0, y que delta\_pic\_order\_cnt[ 1 ] (si se necesita) es igual a 0.

NOTA – El proceso de decodificación deberá suponer una pérdida de imágenes no intencionada cuando se haga referencia a alguno de estos valores de frame\_num que pertenezca a imágenes "inexistente" en el proceso de predicción inter, en las instrucciones de reordenación de listas de imágenes de referencia para imágenes de corto alcance (subcláusula 8.2.4.3.1) o en el proceso de asignación de un LongTermFrameIdx a una imagen de corto alcance (subcláusula 8.2.5.4.3). El proceso de decodificación no deducirá una pérdida de imágenes no intencional cuando se aplique una operación de control de gestión de memoria distinta de 3 a un cuadro marcado como "inexistente".

### 8.2.5.3 Proceso de marcado de imágenes de referencia decodificadas de ventana deslizante

Se llama a este proceso cuando adaptive\_ref\_pic\_marking\_mode\_flag sea igual a 0.

En función de las propiedades de la imagen considerada especificadas a continuación, se aplica lo siguiente:

- Si la imagen considerada es un campo codificado y además es el segundo campo en orden de decodificación de un par de campos de referencias complementarios, y el primer campo ha sido marcado como "utilizado como referencia de corto alcance", la imagen considerada se marca también como "utilizada como referencia de corto alcance".
- De lo contrario, se aplica lo siguiente:
  - Sea numShortTerm el número total de cuadros de referencia, pares de campo de referencia complementarios y campos de referencia desapareados para los cuales hay al menos un campo marcado como "utilizado como referencia de corto alcance". Sea numLongTerm el número total de cuadros de referencia, pares de campo de referencia complementarios y campos de referencia desapareados para los cuales hay al menos un campo marcado como "utilizado como referencia de largo alcance".
  - Cuando numShortTerm + numLongTerm es igual a Max( num\_ref\_frames, 1 ), se deberá cumplir la condición de que numShortTerm sea mayor que 0, y los cuadros de referencia de corto alcance, los pares de campos de referencia complementarios o los campos de referencia desapareados que tienen el valor más pequeño de FrameNumWrap se marcarán como "no utilizados como referencia". Cuando se trata de un cuadro o de un par de campos complementarios, sus dos campos se marcan también como "no utilizados como referencia".

#### **8.2.5.4 Proceso de marcado de imágenes de referencia decodificadas de control adaptativo de memoria**

Se llama a este proceso cuando adaptive\_ref\_pic\_marking\_mode\_flag es igual a 1.

Las instrucciones memory\_management\_control\_operation con valores comprendidos entre 1 y 6 se procesan en el orden en que aparecen en el tren de bits después de que se haya decodificado la imagen considerada. Para cada instrucción memory\_management\_control\_operation, se llama a uno de los procesos especificados en las subcláusulas 8.2.5.4.1 a 8.2.5.4.6 en función del valor de memory\_management\_control\_operation. El valor 0 de la instrucción memory\_management\_control\_operation indica el final de instrucciones memory\_management\_control\_operation.

Las operaciones de control de gestión de memoria se aplican a las imágenes del modo siguiente:

- Si field\_pic\_flag es igual a 0, las instrucciones memory\_management\_control\_operation se aplican a los cuadros o pares de campo de referencia complementarios especificados.
- De lo contrario (field\_pic\_flag es igual a 1), las instrucciones memory\_management\_control\_operation se aplican por separado a los campos de referencias especificados.

##### **8.2.5.4.1 Proceso de marcado de imágenes de corto alcance como "no utilizada como referencia"**

Se llama este proceso cuando memory\_management\_control\_operation es igual a 1.

Sea picNumX dado por:

$$\text{picNumX} = \text{CurrPicNum} - (\text{difference\_of\_pic\_nums\_minus1} + 1). \quad (8-40)$$

En función de field\_pic\_flag el valor de picNumX se utiliza para marcar una imagen de corto alcance como "no utilizada como referencia" del modo siguiente:

- Si field\_pic\_flag es igual a 0, el cuadro de referencia de corto alcance o el par de campos de referencia complementarios de corto alcance especificados por picNumX y sus dos campos se marcan como "no utilizados como referencia".
- De lo contrario (field\_pic\_flag es igual a 1), el campo de referencia de corto alcance especificado por picNumX se marca como "no utilizado como referencia". Cuando el

campo de referencia forma parte de un cuadro de referencia o de un par de campos de referencia complementarios, el cuadro o par de campos complementarios también se marcan como "no utilizado como referencia" aunque el marcado del otro campo no varía.

#### **8.2.5.4.2 Proceso de marcado de imágenes de largo alcance como "no utilizadas para referencia"**

Se llama a este proceso cuando `memory_management_control_operation` es igual a 2.

En función de `field_pic_flag` se utiliza el valor de `LongTermPicNum` para marcar imágenes de largo alcance como "no utilizados como referencia" del modo siguiente:

- Si `field_pic_flag` es igual a 0, el cuadro de referencia de largo alcance o el par de campos de referencia complementarios cuyo `LongTermPicNum` sea igual a `long_term_pic_num` y sus dos campos se marcan como "no utilizados como referencia".
- De lo contrario (`field_pic_flag` es igual a 1), el campo de referencia de largo alcance especificado por `LongTermPicNum` igual a `long_term_pic_num` se marca como "no utilizado como referencia". Cuando el campo de referencia forma parte de un cuadro de referencia o de un par de campos de referencia complementarios, el cuadro o el par de campos complementarios también se marca como "no utilizado como referencia" aunque el marcado del otro campo no varía.

#### **8.2.5.4.3 Proceso de asignación de un LongTermFrameIdx a una imagen de referencia de corto alcance**

Se llama a este proceso cuando `memory_management_control_operation` es igual a 3.

Dado el elemento sintáctico `difference_of_pic_nums_minus1`, la variable `picNumX` se obtiene según se especifica en la subcláusula 8.2.5.4.1. `picNumX` se referirá a un cuadro, par de campos de referencia complementarios o un campo de referencia desapareado marcado como "utilizado como referencia de corto alcance" y que no está marcado como "no existente".

Cuando `LongTermFrameIdx` tenga el valor `long_term_frame_idx` y ya esté asignado a un cuadro de referencia de largo alcance o a un par de campos de referencia complementarios de largo alcance, ese cuadro o par de campos complementarios y sus dos campos se marcan como "no utilizados como referencia". Cuando `LongTermFrameIdx` ya esté asignado a un campo de referencia desapareado y el campo no sea un campo complementario de la imagen especificada por `picNumX`, ese campo se marca como "no utilizado como referencia".

En función de `field_pic_flag` el valor de `LongTermFrameIdx` se utiliza para marcar una imagen "utilizada como referencia de corto alcance" como "utilizada como referencia de largo alcance" del modo siguiente:

- Si `field_pic_flag` es igual a 0, el cuadro de referencia de corto alcance o el par de campos de referencia complementarios de corto alcance especificados por `picNumX` y sus dos campos se cambian de "utilizados como referencia de corto alcance" a "utilizados como referencia de largo alcance" y el valor de `LongTermFrameIdx` se asigna a `long_term_frame_idx`.
- De lo contrario (`field_pic_flag` es igual a 1), el campo de referencia de corto alcance especificado por `picNumX` se cambia de "utilizado como referencia de corto alcance" a "utilizado como referencia de largo alcance" y el valor de `LongTermFrameIdx` se asigna a `long_term_frame_idx`.

#### **8.2.5.4.4 Proceso de decodificación de MaxLongTermFrameIdx**

Se llama a este proceso cuando `memory_management_control_operation` es igual a 4.

Todas las imágenes cuyo `LongTermFrameIdx` es mayor que `max_long_term_frame_idx_plus1 - 1` y que están marcadas como "utilizadas como referencia de largo alcance" se marcarán como "no utilizadas como referencia".



La variable MaxLongTermFrameIdx se calcula del modo siguiente:

- Si max\_long\_term\_frame\_idx\_plus1 es igual a 0, MaxLongTermFrameIdx se pone a "no hay índices de cuadro de largo alcance".
- De lo contrario (max\_long\_term\_frame\_idx\_plus1 es mayor que 0), MaxLongTermFrameIdx se pone a max\_long\_term\_frame\_idx\_plus1 – 1.

NOTA – La instrucción memory\_management\_control\_operation igual a 4 se puede utilizar para marcar imágenes de referencia de largo alcance como "no utilizadas como referencia". En esta Recomendación | Norma Internacional no se especifica la frecuencia de transmisión de max\_long\_term\_frame\_idx\_plus1. Sin embargo el codificador debería enviar una instrucción memory\_management\_control\_operation igual a 4 tras recibir un mensaje de error, como por ejemplo un mensaje petición de regeneración intra.

#### **8.2.5.4.5 Proceso de marcado de todas las imágenes de referencia como "no utilizadas para referencia" y asignación de MaxLongTermFrameIdx a "no hay índices de cuadro de largo alcance"**

Se llama a este proceso cuando memory\_management\_control\_operation es igual a 5.

Todas las imágenes de referencia se marcan como "no utilizadas para referencia" y el valor de la variable MaxLongTermFrameIdx se pone a "no hay índices de cuadro de largo alcance".

#### **8.2.5.4.6 Proceso de asignación de un índice de cuadro de largo alcance a la imagen considerada**

Se llama a este proceso cuando memory\_management\_control\_operation es igual a 6.

Cuando una variable LongTermFrameIdx igual a long\_term\_frame\_idx ya está asignada a un cuadro de referencia de largo alcance o a un par de campos de referencia complementarios de largo alcance, el cuadro o par de campos complementarios y sus dos campos se marcan como "no utilizados como referencia". Cuando LongTermFrameIdx ya está asignado a un campo de referencia desapareado, y el campo no es campo complementario de la imagen considerada, el campo se marca como "no utilizado como referencia".

La imagen considerada se marca como "utilizado como referencia de largo alcance" y LongTermFrameIdx se asigna a long\_term\_frame\_idx.

Cuando field\_pic\_flag es igual a 0, sus dos campos también se marcan como "utilizados como referencia de largo alcance" y LongTermFrameIdx se asigna a long\_term\_frame\_idx.

Cuando field\_pic\_flag es igual a 1 y la imagen considerada es el segundo campo (en orden de decodificación) de un par de campos de referencia complementarios, el par también se marca como "utilizado como referencia de largo alcance" y LongTermFrameIdx se asigna a long\_term\_frame\_idx.

Después de que se marca la imagen de referencia decodificada actual, el número total de cuadros con por lo menos un campo marcado como "utilizado como referencia", más el número de pares de campos complementarios con por lo menos un campo marcado como "utilizado como referencia", más el número de campos desapareados marcados como "utilizado como referencia" no debe ser mayor que Max( num\_ref\_frames, 1).

NOTA – En algunas circunstancias la afirmación anterior puede implicar una restricción sobre el orden en que un elemento sintáctico memory\_management\_control\_operation de valor igual a 6 puede aparecer en la sintaxis de marcación de imágenes de referencia decodificadas con respecto a un elemento sintáctico memory\_management\_control\_operation igual a 1, 2 ó 4.

### **8.3 Proceso de predicción intra**

Se llama a este proceso para macrobloques de tipo I y SI.

Este proceso acepta como argumentos las muestras de macrobloques adyacentes, reconstituidas antes de aplicar el proceso de filtrado de bloques, y en el caso del modo de predicción Intra\_4x4, los valores asociados de Intra4x4PredMode correspondientes a los macrobloques adyacentes.

El resultado de aplicar este proceso es el siguiente:

- Si mb\_type es distinto de I\_PCM, el resultado son las muestras de predicción intra de las componentes del macrobloque, o en caso del proceso de predicción Intra\_4x4 de muestras luma, el resultado son las matrices de muestras luma 4x4 que forman parte de la matriz luma 16x16 de muestras de predicción del macrobloque.
- De lo contrario (mb\_type es igual a I\_PCM) el resultado son las muestras del macrobloque reconstruidas antes de aplicar el proceso de filtrado de bloques.

La variable MvCnt se fija igual a 0.

En función del valor de mb\_type se aplica lo siguiente:

- Si mb\_type es igual a I\_PCM, se llama al proceso descrito en la subcláusula 8.3.4.
- De lo contrario (mb\_type es distinto de I\_PCM) se aplica lo siguiente:
  - El proceso de decodificación para modos de predicción intra se describe para la componentes luma del modo siguiente:
    - si el modo de predicción del macrobloque es igual a intra\_4x4, se aplica la especificación de la subcláusula 8.3.1;
    - de lo contrario (el modo de predicción del macrobloque es igual a Intra\_16x16), se aplica la especificación de la subcláusula 8.3.2.
  - El proceso de decodificación para los modos de predicción intra de componentes croma se describe en la subcláusula 8.3.3.

Las muestras utilizadas en el proceso de predicción intra serán los valores de las muestras antes de aplicar operaciones de filtrado de bloques.

### 8.3.1 Proceso de predicción Intra\_4x4 para muestras luma

Se llama a este proceso cuando el modo de predicción de macrobloques es igual a Intra\_4x4.

Este proceso acepta como argumentos las muestras luma de macrobloques adyacentes reconstituidas ante de aplicar el proceso de filtrado de bloques y los valores asociados de Intra4x4PredMode correspondientes a macrobloques o pares de macrobloques adyacentes.

Este proceso genera como resultado matrices de muestras luma 4x4 que forman parte de la matriz luma 16x16 de muestras de predicción de macrobloque pred<sub>L</sub>.

La componente luma de un macrobloque consta de 16 bloques 4x4 de muestras luma. Estos bloques se barren de manera inversa utilizando el proceso de barrido inverso de bloques luma 4x4 descrito en la subcláusula 6.4.3.

Para todos los bloques luma 4x4 de la componente luma de un macrobloque con luma4x4BlkIdx = 0..15, la variable Intra4x4PredMode[ luma4x4BlkIdx ] se calcula según lo descrito en la subcláusula 8.3.1.1.

Para cada bloque luma de 4x4 muestras indexadas mediante luma4x4BlkIdx = 0..15:

1. Se llama al proceso de predicción de muestras Intra\_4x4 descrito en la subcláusula 8.3.1.2 pasándole como argumentos luma4x4BlkIdx y las muestras reconstituidas antes (en orden de decodificación) de aplicar el proceso de filtrado de bloques correspondientes a bloques luma adyacentes, y el resultado son las muestras de predicción luma Intra\_4x4, pred4x4<sub>L</sub>[ x, y ] con x, y = 0..3.

2. Se calcula la posición de la muestra superior izquierda del bloque luma 4x4 con índice luma4x4BlkIdx dentro del macrobloque considerado, para lo cual se llama al proceso de barrido inverso de bloques luma 4x4 descrito en la subcláusula 6.4.3, pasándole como el argumento luma4x4BlkIdx, y el resultado se asigna a ( xO, yO ) y  $x, y = 0..3$ .

$$\text{pred}_L[ xO + x, yO + y ] = \text{pred}_{4x4}_L[ x, y ] \quad (8-41)$$

3. Se llama al proceso de decodificación de coeficientes de transformada y al proceso de reconstrucción de imágenes antes de aplicar el proceso de filtrado de bloques descrito en la subcláusula 8.5, se le pasan como argumentos  $\text{pred}_L$  y luma4x4BlkIdx, y se obtiene como resultado las muestras reconstituidas correspondientes al bloque luma 4x4 considerado  $S'_L$ .

### 8.3.1.1 Proceso de obtención de Intra4x4PredMode

Este proceso acepta como argumentos el índice del bloque luma 4x4 luma4x4BlkIdx y las matrices variables Intra4x4PredMode que se calculan previamente (en orden de decodificación) de macrobloques adyacentes.

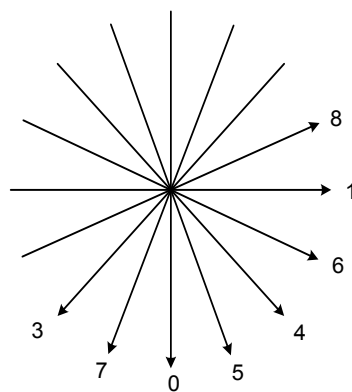
Este proceso genera como resultado la variable  $\text{Intra}_{4x4}\text{PredMode}[ \text{luma}_{4x4}\text{BlkIdx} ]$ .

En el cuadro 8-2 se especifica los valores de  $\text{Intra}_{4x4}\text{PredMode}[ \text{luma}_{4x4}\text{BlkIdx} ]$  y sus correspondientes nombres.

**Cuadro 8-2 – Especificación de  $\text{Intra}_{4x4}\text{PredMode}[ \text{luma}_{4x4}\text{BlkIdx} ]$  y sus correspondientes nombres asociados**

$\text{Intra}_{4x4}\text{PredMode}[ \text{luma}_{4x4}\text{BlkIdx} ]$	Name of $\text{Intra}_{4x4}\text{PredMode}[ \text{luma}_{4x4}\text{BlkIdx} ]$
0	Intra_4x4_Vertical (modo de predicción)
1	Intra_4x4_Horizontal (modo de predicción)
2	Intra_4x4_DC (modo de predicción)
3	Intra_4x4_Diagonal_Down_Left (modo de predicción)
4	Intra_4x4_Diagonal_Down_Right (modo de predicción)
5	Intra_4x4_Vertical_Right (modo de predicción)
6	Intra_4x4_Horizontal_Down (modo de predicción)
7	Intra_4x4_Vertical_Left (modo de predicción)
8	Intra_4x4_Horizontal_Up (modo de predicción)

$\text{Intra}_{4x4}\text{PredMode}[ \text{luma}_{4x4}\text{BlkIdx} ]$  etiquetadas como 0, 1, 3, 4, 5, 6, 7 y 8 representan las direcciones de predicción, según ilustra la figura 8-1.



**Figura 8-1 – Direcciones del modo de predicción Intra\_4x4 (informativo)**

Sea  $\text{intra4x4PredModeA}$  e  $\text{intra4x4PredModeB}$  variables que especifican los modos de predicción Intra de bloques luma 4x4 adyacentes.

$\text{Intra4x4PredMode}[\text{luma4x4BlkIdx}]$  se calcula del modo siguiente:

- Se llama al proceso especificado en la subcláusula 6.4.7.3 pasándole como argumento  $\text{luma4x4BlkIdx}$  y el resultado se asigna a  $\text{mbAddrA}$ ,  $\text{luma4x4BlkIdxA}$ ,  $\text{mbAddrB}$  y  $\text{luma4x4BlkIdxB}$ .
- La variable  $\text{dcOnlyPredicciónFlag}$  se calcula del modo siguiente:
  - si se cumple alguna de las condiciones siguientes,  $\text{dcOnlyPredicciónFlag}$  se pone a 1:
    - el macrobloque con dirección  $\text{mbAddrA}$  no está disponible;
    - el macrobloque con dirección  $\text{mbAddrB}$  no está disponible;
    - el macrobloque con dirección  $\text{mbAddrA}$  está disponible y codificado en modo predicción inter y  $\text{constrained\_intra\_pred\_flag}$  es igual a 1;
    - el macrobloque con dirección  $\text{mbAddrB}$  está disponible y codificado en modo predicción inter y  $\text{constrained\_intra\_pred\_flag}$  es igual a 1;
  - de lo contrario,  $\text{dcOnlyPredicciónFlag}$  se pone a 0.
- Las variables  $\text{intra4x4PredModeN}$ , siendo N A o B, se calcula del modo siguiente:
  - si  $\text{dcOnlyPredictionFlag}$  es igual a 1 o el macrobloque con dirección  $\text{mbAddrN}$  no está codificado en el modo de predicción de macrobloque  $\text{Intra\_4x4}$ ,  $\text{intra4x4PredModeN}$  se pone a 2 (modo de predicción  $\text{Intra\_4x4\_DC}$ );
  - de lo contrario ( $\text{dcOnlyPredictionFlag}$  es igual a 0 y el macrobloque con dirección  $\text{mbAddrN}$  está codificado con el modo de predicción de macrobloques  $\text{Intra\_4x4}$ ),  $\text{intra4x4PredModeN}$  se pone a  $\text{Intra4x4PredMode}[\text{luma4x4BlkIdxN}]$ , siendo  $\text{Intra4x4PredMode}$  la matriz variable asignada al macrobloque  $\text{mbAddrN}$ .
- Para calcular  $\text{Intra4x4PredMode}[\text{luma4x4BlkIdx}]$  se aplica el siguiente procedimiento:

$$\begin{aligned}
 &\text{predIntra4x4PredMode} = \text{Min}(\text{intra4x4PredModeA}, \text{intra4x4PredModeB}) \\
 &\text{if}(\text{prev\_intra4x4\_pred\_mode\_flag}[\text{luma4x4BlkIdx}]) \\
 &\quad \text{Intra4x4PredMode}[\text{luma4x4BlkIdx}] = \text{predIntra4x4PredMode} \\
 &\text{else} \tag{8-42} \\
 &\quad \text{if}(\text{rem\_intra4x4\_pred\_mode}[\text{luma4x4BlkIdx}] < \text{predIntra4x4PredMode}) \\
 &\quad \quad \text{Intra4x4PredMode}[\text{luma4x4BlkIdx}] = \text{rem\_intra4x4\_pred\_mode}[\text{luma4x4BlkIdx}] \\
 &\quad \text{else} \\
 &\quad \quad \text{Intra4x4PredMode}[\text{luma4x4BlkIdx}] = \text{rem\_intra4x4\_pred\_mode}[\text{luma4x4BlkIdx}] + 1
 \end{aligned}$$

### 8.3.1.2 Predicción de muestras $\text{Intra\_4x4}$

Se llama a este proceso para cada bloque luma 4x4 de un macrobloque con modo de predicción a  $\text{Intra\_4x4}$ , seguido del proceso de decodificación de transformada y el proceso de reconstitución de imágenes antes de filtrar cada bloque luma 4x4 luma.

Este proceso acepta como argumentos el índice del bloque luma 4x4 con índice  $\text{luma4x4BlkIdx}$  y las muestras de bloques luma adyacentes reconstituidas antes (en orden de decodificación) de aplicar el proceso el filtrado de bloques.

Este proceso genera como resultado las muestras de predicción  $\text{pred4x4}_l[x, y]$ , con  $x, y = 0..3$  para bloques luma 4x4 con índice  $\text{luma4x4BlkIdx}$ .

Se calcula la posición de la muestra superior izquierda del bloque luma 4x4 con índice  $\text{luma4x4BlkIdx}$  dentro del macrobloque considerado, para lo cual se llama al proceso de barrido inverso de bloques luma 4x4 descrito en la subcláusula 6.4.3, al que se le pasa como argumento  $\text{luma4x4BlkIdx}$ , y el resultado se asigna a  $(xO, yO)$ .

Las 13 muestras adyacentes  $p[x, y]$  que son muestras luma reconstruidas antes de aplicar el proceso de filtrado de bloques, con  $x = -1$ ,  $y = -1..3$  y  $x = 0..7$ ,  $y = -1$ , se calculan del modo siguiente:

- La posición luma (  $x_N, y_N$  ) viene dada por

$$x_N = x_O + x \quad (8-43)$$

$$y_N = y_O + y \quad (8-44)$$

- Se llama al proceso de obtención de posiciones adyacentes descrito en la subcláusula 6.4.8 para posiciones luma, pasándole como argumento (  $x_N, y_N$  ) y se obtiene como resultado  $mbAddrN$  y (  $x_W, y_W$  ).
- Cada muestra  $p[x, y]$  con  $x = -1$ ,  $y = -1..3$  y  $x = 0..7$ ,  $y = -1$  se calcula del modo siguiente:
  - Si se cumple alguna de las condiciones siguientes, la muestra  $p[x, y]$  se marca como "no disponible para predicción Intra\_4x4"
    - $mbAddrN$  no está disponible;
    - el macrobloque  $mbAddrN$  está codificado en modo predicción inter y  $constrained\_intra\_pred\_flag$  es igual a 1;
    - el macrobloque  $mbAddrN$  tiene  $mb\_type$  igual a SI y  $constrained\_intra\_pred\_flag$  es igual a 1 y el macrobloque considerado no tiene  $mb\_type$  igual a SI;
    - $x$  es mayor que 3 y  $luma4x4BlkIdx$  es igual a 3 u 11.
  - De lo contrario, la muestra  $p[x, y]$  se marca como "disponible para predicción Intra\_4x4" y se asigna a  $p[x, y]$  la muestra luma que está en la posición luma (  $x_W, y_W$  ) dentro del macrobloque  $mbAddrN$ .

Cuando las muestras  $p[x, -1]$ , con  $x = 4..7$ , están marcadas como "no disponibles para predicción Intra\_4x4," y la muestra  $p[3, -1]$  está marcada como "disponible para predicción Intra\_4x4," el valor de la muestra de  $p[3, -1]$  se sustituye por los valores de las muestras  $p[x, -1]$ , con  $x = 4..7$ , y las muestras  $p[x, -1]$ , con  $x = 4..7$ , se marcan como "disponible para predicción Intra\_4x4".

NOTA – Se supone que cada bloque está reconstituido dentro de un cuadro antes de decodificar el siguiente bloque.

En función de  $Intra4x4PredMode[luma4x4BlkIdx]$ , se utiliza uno de los modos de predicción Intra\_4x4 descritos en las subcláusulas 8.3.1.2.1 a 8.3.1.2.9.

### 8.3.1.2.1 Especificación del modo de predicción Intra\_4x4\_Vertical

Este modo de predicción Intra\_4x4 se utiliza cuando  $Intra4x4PredMode[luma4x4BlkIdx]$  es igual a 0.

Este modo se utiliza sólo cuando las muestras  $p[x, -1]$ , con  $x = 0..3$ , están marcadas como "disponible para predicción Intra\_4x4".

Los valores de las muestras de predicción  $pred4x4L[x, y]$ , con  $x, y = 0..3$ , serán:

$$pred4x4L[x, y] = p[x, -1], \text{ con } x, y = 0..3 \quad (8-45)$$

### 8.3.1.2.2 Especificación del modo de predicción Intra\_4x4\_Horizontal

Este modo de predicción Intra\_4x4 se utiliza cuando  $Intra4x4PredMode[luma4x4BlkIdx]$  es igual a 1.

Este modo se utiliza solamente cuando las muestras  $p[-1, y]$ , con  $y = 0..3$  están marcadas como "disponible para predicción Intra\_4x4".

Los valores de las muestras de predicción  $pred4x4L[x, y]$ , con  $x, y = 0..3$  serán:

$$pred4x4L[x, y] = p[-1, y], \text{ con } x, y = 0..3 \quad (8-46)$$

### 8.3.1.2.3 Especificación del modo de predicción Intra\_4x4\_DC

Este modo de predicción Intra\_4x4 se utiliza cuando Intra4x4PredMode[ luma4x4BlkIdx ] es igual a 2.

Los valores de las muestras de predicción  $pred4x4_L[x, y]$ , con  $x, y = 0..3$ , se calculan del modo siguiente:

- Si todas las muestras  $p[x, -1]$ , con  $x = 0..3$ , y las muestras  $p[-1, y]$ , con  $y = 0..3$ , están marcadas como "disponible para predicción Intra\_4x4", los valores de las muestras de predicción  $pred4x4_L[x, y]$ , con  $x, y = 0..3$  se calculan mediante la expresión:

$$pred4x4_L[x, y] = (p[0, -1] + p[1, -1] + p[2, -1] + p[3, -1] + p[-1, 0] + p[-1, 1] + p[-1, 2] + p[-1, 3] + 4) \gg 3 \quad (8-47)$$

- De lo contrario, si hay muestras  $p[x, -1]$ , con  $x = 0..3$ , marcadas como "no disponible para predicción Intra\_4x4" y todas las muestras  $p[-1, y]$ , con  $y = 0..3$ , están marcadas como "disponible para predicción Intra\_4x4", los valores de las muestras de predicción  $pred4x4_L[x, y]$ , con  $x, y = 0..3$  se calculan mediante la expresión:

$$pred4x4_L[x, y] = (p[-1, 0] + p[-1, 1] + p[-1, 2] + p[-1, 3] + 2) \gg 2 \quad (8-48)$$

- De lo contrario, si hay muestras  $p[-1, y]$ , con  $y = 0..3$ , marcadas como "no disponible para predicción Intra\_4x4" y todas las muestras  $p[x, -1]$ , con  $x = 0..3$ , están marcadas como "disponible para predicción Intra\_4x4", los valores de las muestras de predicción  $pred4x4_L[x, y]$ , con  $x, y = 0..3$ , se calculan mediante la siguiente expresión:

$$pred4x4_L[x, y] = (p[0, -1] + p[1, -1] + p[2, -1] + p[3, -1] + 2) \gg 2 \quad (8-49)$$

- De lo contrario (algunas muestras  $p[x, -1]$ , con  $x = 0..3$  y algunas muestras  $p[-1, y]$ , con  $y = 0..3$ , están marcadas como "no disponible para predicción Intra\_4x4"), los valores de las muestras de predicción  $pred4x4_L[x, y]$ , con  $x, y = 0..3$ , se calculan mediante la expresión:

$$pred4x4_L[x, y] = 128 \quad (8-50)$$

NOTA – Siempre es posible predecir los bloques luma 4x4 luma utilizando este modo.

### 8.3.1.2.4 Especificación del modo de predicción Intra\_4x4\_Diagonal\_Down\_Left

Este modo de predicción Intra\_4x4 se utiliza cuando Intra4x4PredMode[ luma4x4BlkIdx ] es igual a 3.

Este modo se utilizará solamente cuando las muestras  $p[x, -1]$ , con  $x = 0..7$ , están marcadas como "disponible para predicción Intra\_4x4".

Los valores de las muestras de predicción  $pred4x4_L[x, y]$ , con  $x, y = 0..3$ , se calculan del modo siguiente:

- si  $x$  es igual a 3 e  $y$  es igual a 3,

$$pred4x4_L[x, y] = (p[6, -1] + 3 * p[7, -1] + 2) \gg 2 \quad (8-51)$$

- de lo contrario ( $x$  es distinto de 3, o  $y$  es distinto de 3),

$$pred4x4_L[x, y] = (p[x + y, -1] + 2 * p[x + y + 1, -1] + p[x + y + 2, -1] + 2) \gg 2 \quad (8-52)$$

### 8.3.1.2.5 Especificación del modo de predicción Intra\_4x4\_Diagonal\_Down\_Right

Este modo de predicción Intra\_4x4 se utiliza cuando Intra4x4PredMode[ luma4x4BlkIdx ] es igual a 4.

Este modo se utiliza solamente cuando las muestras  $p[x, -1]$  con  $x = 0..3$  y  $p[-1, y]$ , con  $y = -1..3$ , están marcadas como "disponible para predicción Intra\_4x4".

Los valores de las muestras de predicción  $\text{pred4x4}_L[x, y]$ , con  $x, y = 0..3$ , se calculan del modo siguiente:

– si  $x$  es mayor que  $y$ ,

$$\text{pred4x4}_L[x, y] = (p[x - y - 2, -1] + 2 * p[x - y - 1, -1] + p[x - y, -1] + 2) \gg 2 \quad (8-53)$$

– de lo contrario si  $x$  es menor que  $y$ ,

$$\text{pred4x4}_L[x, y] = (p[-1, y - x - 2] + 2 * p[-1, y - x - 1] + p[-1, y - x] + 2) \gg 2 \quad (8-54)$$

– de lo contrario ( $x$  es igual a  $y$ ),

$$\text{pred4x4}_L[x, y] = (p[0, -1] + 2 * p[-1, -1] + p[-1, 0] + 2) \gg 2 \quad (8-55)$$

### 8.3.1.2.6 Especificación del modo de predicción Intra\_4x4\_Vertical\_Right

Este modo de predicción Intra\_4x4 se utiliza cuando  $\text{Intra4x4PredMode}[\text{luma4x4BlkIdx}]$  es igual a 5.

Este modo se utilizará únicamente cuando las muestras  $p[x, -1]$ , con  $x = 0..3$ , y  $p[-1, y]$ , con  $y = -1..3$ , están marcadas como "disponible para predicción Intra\_4x4".

Sea  $zVR$  una variable cuyo valor es  $2 * x - y$ .

Los valores de las muestras de predicción  $\text{pred4x4}_L[x, y]$ , con  $x, y = 0..3$ , se calculan del modo siguiente:

– si  $zVR$  es igual a 0, 2, 4, ó 6,

$$\text{pred4x4}_L[x, y] = (p[x - (y \gg 1) - 1, -1] + p[x - (y \gg 1), -1] + 1) \gg 1 \quad (8-56)$$

– de lo contrario, si  $zVR$  es igual a 1, 3 ó 5,

$$\text{pred4x4}_L[x, y] = (p[x - (y \gg 1) - 2, -1] + 2 * p[x - (y \gg 1) - 1, -1] + p[x - (y \gg 1), -1] + 2) \gg 2 \quad (8-57)$$

– de lo contrario, si  $zVR$  es igual a  $-1$ ,

$$\text{pred4x4}_L[x, y] = (p[-1, 0] + 2 * p[-1, -1] + p[0, -1] + 2) \gg 2 \quad (8-58)$$

– de lo contrario ( $zVR$  es igual a  $-2$  ó  $-3$ ),

$$\text{pred4x4}_L[x, y] = (p[-1, y - 1] + 2 * p[-1, y - 2] + p[-1, y - 3] + 2) \gg 2 \quad (8-59)$$

### 8.3.1.2.7 Especificación del modo de predicción Intra\_4x4\_Horizontal\_Down

Este modo de predicción Intra\_4x4 se utiliza cuando  $\text{Intra4x4PredMode}[\text{luma4x4BlkIdx}]$  es igual a 6.

Este modo se utilizará únicamente cuando las muestras  $p[x, -1]$ , con  $x = 0..3$ , y  $p[-1, y]$ , con  $y = -1..3$ , están marcadas como "disponible para predicción Intra\_4x4".

Sea  $zHD$  la variable cuyo valor es  $2 * y - x$ .

Los valores de las muestras de predicción  $\text{pred4x4}_L[x, y]$ , con  $x, y = 0..3$ , se calculan del modo siguiente:

– si  $zHD$  es igual a 0, 2, 4 ó 6,

$$\text{pred4x4}_L[x, y] = (p[-1, y - (x \gg 1) - 1] + p[-1, y - (x \gg 1)] + 1) \gg 1 \quad (8-60)$$

– de lo contrario, si  $zHD$  es igual a 1, 3, ó 5,

$$\text{pred4x4}_L[x, y] = (p[-1, y - (x \gg 1) - 2] + 2 * p[-1, y - (x \gg 1) - 1] + p[-1, y - (x \gg 1)] + 2) \gg 2 \quad (8-61)$$

– de lo contrario, si  $zHD$  es igual a  $-1$ ,

$$\text{pred4x4}_L[x, y] = (p[-1, 0] + 2 * p[-1, -1] + p[0, -1] + 2) \gg 2 \quad (8-62)$$

– de lo contrario ( $zHD$  es igual a  $-2$  ó  $-3$ ),

$$\text{pred4x4}_L[x, y] = (p[x - 1, -1] + 2 * p[x - 2, -1] + p[x - 3, -1] + 2) \gg 2 \quad (8-63)$$

### 8.3.1.2.8 Especificación del modo de predicción Intra\_4x4\_Vertical\_Left

Este modo de predicción Intra\_4x4 se utiliza cuando Intra4x4PredMode[ luma4x4BlkIdx ] es igual a 7.

Este modo se utiliza únicamente cuando las muestras  $p[x, -1]$ , con  $x = 0..7$ , están marcadas como "disponible para predicción Intra\_4x4".

Los valores de la muestra de predicción  $\text{pred4x4}_L[x, y]$ , con  $x, y = 0..3$ , se calculan del modo siguiente:

- si  $y$  es igual a 0 ó 2,

$$\text{pred4x4}_L[x, y] = (p[x + (y \gg 1), -1] + p[x + (y \gg 1) + 1, -1] + 1) \gg 1 \quad (8-64)$$

- de lo contrario ( $y$  es igual a 1 ó 3),

$$\text{pred4x4}_L[x, y] = (p[x + (y \gg 1), -1] + 2 * p[x + (y \gg 1) + 1, -1] + p[x + (y \gg 1) + 2, -1] + 2) \gg 2 \quad (8-65)$$

### 8.3.1.2.9 Especificación del modo de predicción Intra\_4x4\_Horizontal\_Up

Este modo de predicción Intra\_4x4 se utiliza cuando Intra4x4PredMode[ luma4x4BlkIdx ] es igual a 8.

Este modo se utiliza únicamente cuando las muestras  $p[-1, y]$ , con  $y = 0..3$ , están marcadas como "disponible para predicción Intra\_4x4".

Sea  $zHU$  la variable cuyo valor es  $x + 2 * y$ .

Los valores de la muestra de predicción  $\text{pred4x4}_L[x, y]$ , con  $x, y = 0..3$ , se calcula del modo siguiente:

- si  $zHU$  es igual a 0, 2, ó 4

$$\text{pred4x4}_L[x, y] = (p[-1, y + (x \gg 1)] + p[-1, y + (x \gg 1) + 1] + 1) \gg 1 \quad (8-66)$$

- de lo contrario, si  $zHU$  es igual a 1 ó 3

$$\text{pred4x4}_L[x, y] = (p[-1, y + (x \gg 1)] + 2 * p[-1, y + (x \gg 1) + 1] + p[-1, y + (x \gg 1) + 2] + 2) \gg 2 \quad (8-67)$$

- dDe lo contrario, si  $zHU$  es igual a 5,

$$\text{pred4x4}_L[x, y] = (p[-1, 2] + 3 * p[-1, 3] + 2) \gg 2 \quad (8-68)$$

- de lo contrario ( $zHU$  es mayor que 5),

$$\text{pred4x4}_L[x, y] = p[-1, 3] \quad (8-69)$$

## 8.3.2 Proceso de predicción Intra\_16x16 para muestras luma

Se llama a este proceso cuando el modo de predicción de macrobloque es igual a Intra\_16x16. Este proceso especifica la forma de calcular las muestras luma de predicción Intra para el macrobloque considerado.

Ese proceso acepta como argumentos las muestras reconstruidas de los bloques luma (si los hubiere) antes de aplicar un proceso de filtrado de bloques correspondiente.

Este proceso genera como resultado las muestras luma de predicción Intra del macrobloque considerado  $\text{pred}_L[x, y]$ .

Las 33 muestras adyacentes  $p[x, y]$  que son las muestras luma reconstruidas antes de aplicar el proceso de filtrado de bloques, con  $x = -1, y = -1..15$  y con  $x = 0..15, y = -1$ , se calculan del modo siguiente:



- Se llama al proceso de cálculo de posiciones adyacentes descrito en la subcláusula 6.4.8, pasándole como argumentos las posiciones luma ( x, y ) igual a ( xN, yN ), y se obtiene como resultado mbAddrN y ( xW, yW ).
- Cada muestra p[ x, y ] con x = -1, y = -1..15 y con x = 0..15, y = -1 se calcula del modo siguiente:
  - Si se cumple alguna de las condiciones siguientes, la muestra p[ x, y ] se marca como "no disponible para predicción Intra\_16x16":
    - mbAddrN no está disponible;
    - la mbAddrN del macrobloque está codificada en modo de predicción Inter y constrained\_intra\_pred\_flag es igual a 1;
    - la mbAddrN del macrobloque tiene mb\_type igual a SI y constrained\_intra\_pred\_flag es igual a 1.
  - De lo contrario, la muestra p[ x, y ] se marca como "disponible para predicción Intra\_16x16" y se asigna a p[ x, y ] la muestra luma en la posición luma ( xW, yW ) dentro de la mbAddrN.

Sean pred<sub>L</sub>[ x, y ], con x, y = 0..15, las muestras de predicción de las muestras del bloque luma 16x16.

En el cuadro 8-3 se especifican los modos de predicción Intra\_16x16.

**Cuadro 8-3 – Especificación de Intra16x16PredMode y sus correspondientes nombres**

Intra16x16PredMode	Name of Intra16x16PredMode
0	Intra_16x16_Vertical (modo de predicción)
1	Intra_16x16_Horizontal (modo de predicción)
2	Intra_16x16_DC (modo de predicción)
3	Intra_16x16_Plane (modo de predicción)

En función de Intra16x16PredMode, se utiliza uno de los modos de predicción Intra\_16x16 especificados en las subcláusulas 8.3.2.1 a 8.3.2.4.

### 8.3.2.1 Especificación del modo de predicción Intra\_16x16\_Vertical

Este modo de predicción Intra\_16x16 se utiliza sólo cuando las muestras p[ x, -1 ] con x = 0..15 están marcadas como "disponibles para predicción Intra\_16x16".

$$\text{pred}_L[x, y] = p[x, -1], \text{ con } x, y = 0..15 \quad (8-70)$$

### 8.3.2.2 Especificación del modo de predicción Intra\_16x16\_Horizontal

Este modo de predicción Intra\_16x16 se utiliza únicamente cuando las muestras p[-1, y] con y = 0..15 están marcadas como "disponible para predicción Intra\_16x16".

$$\text{pred}_L[x, y] = p[-1, y], \text{ con } x, y = 0..15 \quad (8-71)$$

### 8.3.2.3 Especificación del modo de predicción Intra\_16x16\_DC

Este modo de predicción Intra\_16x16 se utilizará en función de si las muestras adyacentes están marcadas como "disponible para predicción Intra\_16x16" del modo siguiente:

- Si todas las muestras adyacentes p[ x, -1 ], donde x = 0..15 y p[ -1, y ], donde y = 0..15 están marcadas como "disponibles para Intra\_16x16", la predicción para todas las muestras luma en el macrobloque viene dada por la expresión:

$$\text{pred}_L[x, y] = \left( \sum_{x'=0}^{15} p[x', -1] + \sum_{y'=0}^{15} p[-1, y'] + 16 \right) \gg 5, \text{ con } x, y = 0..15 \quad (8-72)$$

- De lo contrario, si alguna de las muestras adyacentes  $p[x, -1]$ , donde  $x = 0..15$ , está marcada como "no disponible para predicción Intra\_16x16" y todas las muestras adyacentes  $p[-1, y]$ , donde  $y = 0..15$ , están marcadas como "disponible para predicción Intra\_16x16", la predicción de todas las muestras luma viene dada por la expresión:

$$\text{pred}_L[x, y] = \left( \sum_{y'=0}^{15} p[-1, y'] + 8 \right) \gg 4, \text{ con } x, y = 0..15 \quad (8-73)$$

- De lo contrario, si alguna de las muestras adyacentes  $p[-1, y]$ , donde  $y = 0..15$ , está marcada como "disponible para predicción Intra\_16x16" y todas las muestras adyacentes  $p[x, -1]$ , donde  $x = 0..15$ , están marcadas como "disponible para predicción Intra\_16x16", la predicción de todas las muestras luma en el macrobloque viene dada por la expresión:

$$\text{pred}_L[x, y] = \left( \sum_{x'=0}^{15} p[x', -1] + 8 \right) \gg 4 \text{ con } x, y = 0..15 \quad (8-74)$$

- De lo contrario (algunas de las muestras adyacentes  $p[x, -1]$ , donde  $x = 0..15$ , y algunas de las muestras adyacentes  $p[-1, y]$ , donde  $y = 0..15$ , están marcadas como "no disponible para predicción Intra\_16x16"), la predicción de todas las muestras luma en el macrobloque viene dada por la expresión:

$$\text{pred}_L[x, y] = 128, \text{ con } x, y = 0..15 \quad (8-75)$$

### 8.3.2.4 Especificación del modo de predicción en Intra\_16x16\_Plane

Este modo de predicción Intra\_16x16 se utilizará únicamente cuando las muestras  $p[x, -1]$  con  $x = -1..15$  y  $p[-1, y]$  con  $y = 0..15$  están marcada como "disponible para predicción Intra\_16x16".

$$\text{pred}_L[x, y] = \text{Clip1}((a + b * (x - 7) + c * (y - 7) + 16) \gg 5), \text{ con } x, y = 0..15, \quad (8-76)$$

siendo:

$$a = 16 * (p[-1, 15] + p[15, -1]) \quad (8-77)$$

$$b = (5 * H + 32) \gg 6 \quad (8-78)$$

$$c = (5 * V + 32) \gg 6 \quad (8-79)$$

y H y V vienen dadas por las ecuaciones 8-80 y 8-81.

$$H = \sum_{x'=0}^7 (x'+1) * (p[8+x', -1] - p[6-x', -1]) \quad (8-80)$$

$$V = \sum_{y'=0}^7 (y'+1) * (p[-1, 8+y'] - p[-1, 6-y']) \quad (8-81)$$

### 8.3.3 Proceso de predicción intra para muestras croma

Se llama a este proceso para macrobloques de los tipos I y SI. Este proceso especifica la manera de calcular las muestras croma de predicción intra para el macrobloque considerado.

Este proceso acepta como argumentos las muestras reconstruidas de los bloques croma adyacentes (si los hubiese) antes de aplicar el proceso de filtrado.

Este proceso genera como resultado las muestras croma de predicción intra del macrobloque considerado  $\text{pred}_{Cb}[x, y]$  y  $\text{pred}_{Cr}[x, y]$ .

Los dos bloques croma (Cb y Cr) del macrobloque utilizarán el mismo modo de predicción. El modo de predicción se aplica por separado a cada bloque croma. Se llama al proceso descrito en esta subcláusula para cada bloque croma. En el resto de esta subcláusula, por bloque croma se

entiende uno de los dos bloques croma y se utiliza el subíndice C para referirse a ambos índices Cb y Cr.

Las 17 muestras adyacentes  $p[x, y]$  que son muestras croma reconstruidas antes de aplicar el proceso de filtrado de bloques, con  $x = -1, y = -1..7$  y con  $x = 0..7, y = -1$ , se calculan del modo siguiente:

- Se llama al proceso de cálculo de posiciones adyacentes descrito en la subcláusula 6.4.8, pasándole como argumentos las posiciones croma  $(x, y)$  igual a  $(xN, yN)$ , y se obtiene como resultado  $mbAddrN$  y  $(xW, yW)$ .
- Cada muestra  $p[x, y]$  se obtiene del modo siguiente:
  - Si se cumple algunas de las condiciones siguientes, la muestra  $p[x, y]$  se marca como "no disponible para predicción intra croma":
    - $mbAddrN$  no está disponible;
    - el macrobloque  $mbAddrN$  esta codificado en modo predicción inter y  $constrained\_intra\_pred\_flag$  es igual a 1;
    - el macrobloque  $mbAddrN$  tiene  $mb\_type$  igual a SI y  $constrained\_intra\_pred\_flag$  es igual a 1 y el macrobloque considerado no tiene  $mb\_type$  igual a SI.
  - De lo contrario la muestra  $p[x, y]$  está marcada como "disponible para predicción intra croma" y se asigna a  $p[x, y]$  la muestra croma de componente C en la posición croma  $(xW, yW)$  dentro del macrobloque  $mbAddrN$ .

Sea  $pred_c[x, y]$  con  $x, y = 0..7$  las muestras de predicción de las muestras del bloque croma.

En el cuadro 8-4 se especifican los modos de predicción intra croma.

**Cuadro 8-4 – Especificación de los modos de predicción intra croma y sus correspondientes nombres**

<b>intra_chroma_pred_mode</b>	<b>Nombre del intra_chroma_pred_mode</b>
0	Intra_Chroma_DC (modo de predicción)
1	Intra_Chroma_Horizontal (modo de predicción)
2	Intra_Chroma_Vertical (modo de predicción)
3	Intra_Chroma_Plane (modo de predicción)

En función de  $intra\_chroma\_pred\_mode$ , se utiliza uno de los modos de predicción intra croma descritos en las subcláusulas 8.3.3.1 a 8.3.3.4.

### 8.3.3.1 Especificación del modo de predicción Intra\_Chroma\_DC

Los valores de las muestras de predicción  $pred_c[x, y]$ , con  $x = 0..3$  e  $y = 0..3$ , se calculan del modo siguiente:

- Si todas las muestras  $p[x, -1]$ , con  $x = 0..3$  y todas las muestras  $p[-1, y]$ , con  $y = 0..3$  están marcadas como "disponible para predicción intra croma",

$$pred_c[x, y] = \left( \sum_{x'=0}^3 p[x', -1] + \sum_{y'=0}^3 p[-1, y'] + 4 \right) \gg 3, \text{ con } x = 0..3 \text{ e } y = 0..3 \quad (8-82)$$

- De lo contrario, si todas las muestras  $p[x, -1]$  con  $x = 0..3$  están marcadas como, "disponible para predicción intra croma" y ninguna muestra  $p[-1, y]$  con  $y = 0..3$  está marcada como "no disponible para predicción intra croma",

$$\text{pred}_C[x, y] = \left( \sum_{x'=0}^3 p[x', -1] + 2 \right) \gg 2, \text{ con } x = 0..3 \text{ e } y = 0..3 \quad (8-83)$$

- De lo contrario, si ninguna muestra  $p[x, -1]$  con  $x = 0..3$  está marcada como "no disponible para predicción intra cromas" y todas muestras  $p[-1, y]$  con  $y = 0..3$  están marcadas como "disponible para predicción intra cromas",

$$\text{pred}_C[x, y] = \left( \sum_{y'=0}^3 p[-1, y'] + 2 \right) \gg 2, \text{ con } x = 0..3 \text{ e } y = 0..3 \quad (8-84)$$

- De lo contrario (algunas muestras  $p[x, -1]$  con  $x = 0..3$  y algunas muestras  $p[-1, y]$  con  $y = 0..3$  están marcadas como "no disponible para predicción intra cromas"),

$$\text{pred}_C[x, y] = 128, \text{ con } x = 0..3 \text{ e } y = 0..3 \quad (8-85)$$

Los valores de las muestras de predicción  $\text{pred}_C[x, y]$  con  $x = 4..7$  e  $y = 0..3$  se calculan del modo siguiente:

- Si todas las muestras  $p[x, -1]$  con  $x = 4..7$  están marcadas como "disponible para predicción intra cromas",

$$\text{pred}_C[x, y] = \left( \sum_{x'=4}^7 p[x', -1] + 2 \right) \gg 2, \text{ con } x = 4..7 \text{ e } y = 0..3 \quad (8-86)$$

- De lo contrario, si todas muestras  $p[-1, y]$  con  $y = 0..3$  están marcadas como "disponible para predicción intra cromas",

$$\text{pred}_C[x, y] = \left( \sum_{y'=0}^3 p[-1, y'] + 2 \right) \gg 2, \text{ con } x = 4..7 \text{ e } y = 0..3 \quad (8-87)$$

- De lo contrario (algunas muestras  $p[x, -1]$  con  $x = 4..7$  y algunas muestras  $p[-1, y]$  con  $y = 0..3$  están marcadas como "no disponible para predicción intra cromas"),

$$\text{pred}_C[x, y] = 128, \text{ con } x = 4..7 \text{ e } y = 0..3 \quad (8-88)$$

Los valores de las muestras  $\text{pred}_C[x, y]$  con  $x = 0..3$  e  $y = 4..7$  se calculan del modo siguiente:

- Si todas las muestras  $p[-1, y]$  con  $y = 4..7$  están marcadas como "disponible para predicción intra cromas",

$$\text{pred}_C[x, y] = \left( \sum_{y'=4}^7 p[-1, y'] + 2 \right) \gg 2, \text{ con } x = 0..3 \text{ e } y = 4..7 \quad (8-89)$$

- De lo contrario, si todas las muestras  $p[x, -1]$  con  $x = 0..3$  están marcadas como "disponible para predicción intra cromas",

$$\text{pred}_C[x, y] = \left( \sum_{x'=0}^3 p[x', -1] + 2 \right) \gg 2, \text{ con } x = 0..3 \text{ e } y = 4..7 \quad (8-90)$$

- De lo contrario (algunas muestras  $p[x, -1]$  con  $x = 0..3$  y algunas muestras  $p[-1, y]$  con  $y = 4..7$  están marcadas como "no disponible para predicción intra cromas"),

$$\text{pred}_C[x, y] = 128, \text{ con } x = 0..3 \text{ e } y = 4..7 \quad (8-91)$$

Los valores de las muestras de predicción  $\text{pred}_C[x, y]$  con  $x = 4..7$  e  $y = 4..7$  se calculan del modo siguiente:

- Si todas muestras  $p[x, -1]$  con  $x = 4..7$  y todas muestras  $p[-1, y]$  con  $y = 4..7$  están marcadas como "disponible para predicción intra cromas",

$$\text{pred}_C[x, y] = \left( \sum_{x'=4}^7 p[x', -1] + \sum_{y'=4}^7 p[-1, y'] + 4 \right) \gg 3, \text{ con } x = 4..7 \text{ e } y = 4..7 \quad (8-92)$$

- De lo contrario, si todas las muestras  $p[x, -1]$  con  $x = 4..7$  están marcadas como "disponible para predicción intra cromas" y ninguna muestra  $p[-1, y]$  con  $y = 4..7$  está marcada como "no disponible para predicción intra cromas",

$$\text{pred}_C[x, y] = \left( \sum_{x'=4}^7 p[x', -1] + 2 \right) \gg 2, \text{ con } x = 4..7 \text{ e } y = 4..7 \quad (8-93)$$

- De lo contrario, si ninguna muestra  $p[x, -1]$ , con  $x = 4..7$  está marcada como "no disponible para predicción intra cromas" y todas las muestras  $p[-1, y]$  con  $y = 4..7$  están marcadas como "disponible para predicción intra cromas",

$$\text{pred}_C[x, y] = \left( \sum_{y'=4}^7 p[-1, y'] + 2 \right) \gg 2, \text{ con } x = 4..7 \text{ e } y = 4..7 \quad (8-94)$$

- De lo contrario (algunas muestras  $p[x, -1]$ , con  $x = 4..7$  y algunas muestras  $p[-1, y]$  con  $y = 4..7$  están marcadas como "no disponible para predicción intra cromas"),

$$\text{pred}_C[x, y] = 128, \text{ con } x = 4..7 \text{ e } y = 4..7 \quad (8-95)$$

### 8.3.3.2 Especificación del modo predicción Intra\_Chroma\_Horizontal

Este modo se utilizará únicamente cuando las muestras  $p[-1, y]$  con  $y = 0..7$  están marcadas como "disponible para predicción intra cromas".

Los valores de las muestras de predicción  $\text{pred}_C[x, y]$  se calculan del modo siguiente:

$$\text{pred}_C[x, y] = p[-1, y], \text{ con } x, y = 0..7 \quad (8-96)$$

### 8.3.3.3 Especificación del modo de predicción Intra\_Chroma\_Vertical

Este modo se utilizará únicamente cuando las muestras  $p[x, -1]$  con  $x = 0..7$ , están marcadas como "disponible para predicción intra cromas".

Los valores de las muestras de predicción  $\text{pred}_C[x, y]$  se calculan del modo siguiente:

$$\text{pred}_C[x, y] = p[x, -1], \text{ con } x, y = 0..7 \quad (8-97)$$

### 8.3.3.4 Especificación del modo de predicción Intra\_Chroma\_Plane

Este modo se utilizará únicamente cuando las muestras  $p[x, -1]$ , con  $x = 0..7$  y  $p[-1, y]$  con  $y = -1..7$  están marcadas como "disponible para predicción intra cromas".

Los valores de las muestras de predicción  $\text{pred}_C[x, y]$  se calculan del modo siguiente:

$$\text{pred}_C[x, y] = \text{Clip1}((a + b * (x - 3) + c * (y - 3) + 16) \gg 5), \text{ con } x, y = 0..7 \quad (8-98)$$

siendo:

$$a = 16 * (p[-1, 7] + p[7, -1]) \quad (8-99)$$

$$b = (17 * H + 16) \gg 5 \quad (8-100)$$

$$c = (17 * V + 16) \gg 5 \quad (8-101)$$

y H y V se definen así:

$$H = \sum_{x'=0}^3 (x'+1) * (p[4 + x', -1] - p[2 - x', -1]) \quad (8-102)$$

$$V = \sum_{y'=0}^3 (y'+1) * (p[-1, 4 + y'] - p[-1, 2 - y']) \quad (8-103)$$

### 8.3.4 Proceso de reconstrucción de muestras para macrobloques I\_PCM

Se llama a este proceso cuando `mb_type` es igual a `I_PCM`.

Este proceso genera como resultado las muestras de macrobloques reconstruidas  $S'_L$ ,  $S'_{Cb}$  y  $S'_{Cr}$  antes de aplicar el proceso de filtrado de bloques.

La variable `dy` se calcula del modo siguiente:

- Si `MbaffFrameFlag` es igual a 1 y el macrobloque considerado es un macrobloque campo, `dy` se pone a 2.
- De lo contrario (`MbaffFrameFlag` es igual a 0 o el macrobloque considerado es un macrobloque cuadro), `dy` se pone a 1.

Para calcular la posición de la muestra luma superior izquierda del macrobloque considerado se llama al proceso de barrido inverso de macrobloque descrito en la subcláusula 6.4.1, pasándole como argumento `CurrMbAddr` y el resultado se asigna a  $(xP, yP)$ .

Las muestras reconstruidas antes de aplicar el proceso de filtrado de bloques se generan del modo siguiente:

```
para( i = 0; i < 256; i++ )
     $S'_L[ xP + ( i \% 16 ), yP + dy * ( i / 16 ) ] = pcm\_byte[ i ]$  (8-104)
```

```
para( i = 0; i < 64; i++ ) {
     $S'_{Cb}[ ( xP >> 1 ) + ( i \% 8 ), ( ( yP + 1 ) >> 1 ) + dy * ( i / 8 ) ] = pcm\_byte[ i + 256 ]$  (8-105)
     $S'_{Cr}[ ( xP >> 1 ) + ( i \% 8 ), ( ( yP + 1 ) >> 1 ) + dy * ( i / 8 ) ] = pcm\_byte[ i + 320 ]$ 
}
```

### 8.4 Proceso de predicción inter

Se llama a este proceso al decodificar macrobloques de los tipos P y B.

Este proceso genera como resultado las muestras de predicción inter del macrobloque considerado, que son una matriz  $16 \times 16$   $pred_L$  de muestras luma y dos matrices  $8 \times 8$   $pred_{Cr}$  y  $pred_{Cb}$  de muestras cromas, una para cada componente cromas `Cb` y `Cr`.

La partición del macrobloque se especifica por `mb_type`. A cada partición macrobloque se hace referencia mediante `mbPartIdx`. Cuando la partición macrobloque consta de particiones que son iguales a submacrobloques, cada submacrobloque se puede subdividir nuevamente en particiones submacrobloques, según lo especifique `sub_mb_type`. A cada partición submacrobloque se hace referencia mediante `subMbPartIdx`. Cuando las particiones macrobloque no están formadas por submacrobloques, `subMbPartIdx` se pone a 0.

Para cada partición macrobloque o partición submacrobloque se aplican los siguientes pasos.

Las funciones `MbPartWidth()`, `MbPartHeight()`, `SubMbPartWidth()` y `SubMbPartHeight()` que describen la anchura y altura de las particiones macrobloque y particiones submacrobloque se especifican en los cuadros 7-10, 7-11, 7-14 y 7-15.

Las variables `partWidth` y `partHeight` se calculan del modo siguiente:

- Si `mb_type` es distinto de `P_8x8` o `P_8x8ref0` o `B_8x8`, se aplica lo siguiente:  
 $partWidth = MbPartWidth( mb\_type )$  (8-106)

$partHeight = MbPartHeight( mb\_type )$  (8-107)

- De lo contrario (`mb_type` es igual a `P_8x8` o `P_8x8ref0` o `B_8x8`):  
 $partWidth = SubMbPartWidth( sub\_mb\_type[ mbPartIdx ] )$  (8-108)  
 $partHeight = SubMbPartHeight( sub\_mb\_type[ mbPartIdx ] )$  (8-109)

Cuando `mb_type` es igual a `B_Skip` o `B_Direct_16x16` o `sub_mb_type[ mbPartIdx ]` es igual a `B_Direct_8x8`, el proceso de predicción Inter se define para:

partWidth = 4 (8-110)

partHeight = 4 (8-111)

donde mbPartIdx puede tomar los valores 0..3. Para cada submacrobloque indexado por mbPartIdx, subMbPartIdx puede tomar los valores 0..3.

Fíjese la variable MvCnt inicialmente igual a cero antes de cualquier invocación de la subcláusula 8.4.1 para el macrobloque.

El proceso de predicción inter de una partición macrobloque mbPartIdx y de una partición submacrobloque subMbPartIdx consta de los siguientes pasos.

1. Cálculo de las componentes del vector de movimiento y de índices de referencia, según se define en la subcláusula 8.4.1.

Este proceso acepta como argumentos:

- una partición macrobloque mbPartIdx,
- una partición submacrobloque subMbPartIdx.

Este proceso genera como resultado:

- los vectores de movimiento luma mvL0 y mvL1 y los vectores de movimiento croma mvCL0 y mvCL1,
- los índices de referencia refIdxL0 y refIdxL1,
- los indicadores de utilización de listas de predicción predFlagL0 y predFlagL1.
- el conteo de vectores de movimiento de la partición de submacrobloque, subMvCnt.

2. La variable MvCnt se incrementa en subMvCnt.

3. Proceso de decodificación para el cálculo de las muestras de interpredicción, según lo especifica la subcláusula 8.4.2.

Este proceso acepta como argumentos:

- una partición macrobloque mbPartIdx,
- una partición submacrobloque subMbPartIdx,
- las variables anchura y altura de la partición, partWidth y partHeight,
- los vectores de movimiento luma mvL0 y mvL1 y los vectores de movimiento croma mvCL0 y mvCL1,
- los índices de referencia refIdxL0 y refIdxL1,
- los indicadores de utilización de lista de predicción predFlagL0 y predFlagL1.

Este proceso genera como resultado:

- las muestras de predicción inter (pred), que son una matriz (partWidth)x(partHeight) predPart<sub>L</sub> de muestras luma de predicción y dos matrices (partWidth/2)x(partHeight/2) predPart<sub>Cr</sub> y predPart<sub>Cb</sub> de muestras croma de predicción, una para cada componente croma, Cb y Cr.

Se realizan las siguientes asignaciones que se utilizarán más adelante en el proceso de decodificación para calcular variables:

MvL0[ mbPartIdx ][ subMbPartIdx ] = mvL0 (8-112)

MvL1[ mbPartIdx ][ subMbPartIdx ] = mvL1 (8-113)

RefIdxL0[ mbPartIdx ] = refIdxL0 (8-114)

RefIdxL1[ mbPartIdx ] = refIdxL1 (8-115)

PredFlagL0[ mbPartIdx ] = predFlagL0 (8-116)

$$\text{PredFlagL1}[\text{mbPartIdx}] = \text{predFlagL1} \quad (8-117)$$

Se calcula la posición de la muestra superior izquierda de la partición relativa a la muestra superior izquierda del macrobloque para lo cual se llama al proceso de barrido inverso de particiones macrobloque descrito en la subcláusula 6.4.2.1, al que se le pasa como argumento `mbPartIdx` y el resultado se asigna a (`xP`, `yP`).

Se calcula la posición de la muestra superior izquierda de la subpartición macrobloque relativa a la muestra superior izquierda de la partición macrobloque, para lo cual se llama al proceso de barrido inverso de particiones submacrobloque descrito en la subcláusula 6.4.2.2, al que se le pasa como argumento `subMbPartIdx` y el resultado se asigna a (`xS`, `yS`).

Para formar la predicción del macrobloque, las muestras de predicción de la partición o de la partición submacrobloque se sitúan en sus posiciones relativas correctas en el macrobloque, del siguiente modo.

Se calcula la variable  $\text{pred}_L[\text{xP} + \text{xS} + \text{x}, \text{yP} + \text{yS} + \text{y}]$ , con  $\text{x} = 0 \dots \text{partWidth} - 1$ ,  $\text{y} = 0 \dots \text{partHeight} - 1$  mediante la expresión:

$$\text{pred}_L[\text{xP} + \text{xS} + \text{x}, \text{yP} + \text{yS} + \text{y}] = \text{predPart}_L[\text{x}, \text{y}] \quad (8-118)$$

Se calcula la variable  $\text{pred}_C[\text{xP} / 2 + \text{xS} / 2 + \text{x}, \text{yP} / 2 + \text{yS} / 2 + \text{y}]$ , con  $\text{x} = 0 \dots \text{partWidth}/2 - 1$ ,  $\text{y} = 0 \dots \text{partHeight}/2 - 1$ , siendo `C` igual a `Cb` o `Cr`, mediante la expresión:

$$\text{pred}_C[\text{xP} / 2 + \text{xS} / 2 + \text{x}, \text{yP} / 2 + \text{yS} / 2 + \text{y}] = \text{predPart}_C[\text{x}, \text{y}] \quad (8-119)$$

#### 8.4.1 Proceso de cálculo de las componentes de vector de movimiento e índices de referencia

Este proceso acepta como argumentos:

- una partición de macrobloque `mbPartIdx`,
- una partición de submacrobloque `subMbPartIdx`.

Este proceso genera como resultado:

- los vectores de movimiento `luma`, `mvL0` y `mvL1`, y los vectores de movimiento `croma`, `mvCL0` y `mvCL1`,
- los índices de referencia `refIdxL0` y `refIdxL1`,
- los indicadores de utilización de listas de predicción, `predFlagL0` y `predFlagL1`,
- una variable para el conteo de vectores de movimiento de macrobloques de subparticiones, `subMvCnt`.

Para calcular las variables `mvL0` y `mvL1`, y `refIdxL0` y `refIdxL1`, se aplica lo siguiente:

- Si `mb_type` es igual a `P_Skip`, se llama al proceso de cálculo de vectores de movimiento `luma` para macrobloques obviados en sectores `P` y `SP`, descrito en la subcláusula 8.4.1.1, el cual genera como resultado los vectores de movimiento `luma` `mvL0` y los índices de referencia `refIdxL0`, y además `predFlagL0` se pone a 1. `mvL1` y `refIdxL1` se marcan como no disponibles y `predFlagL1` se pone a 0. La variable para el conteo de vectores de movimiento de subpartición `subMvCnt` se pone igual a 1.
- De lo contrario, si `mb_type` es igual a `B_Skip`, o `B_Direct_16x16` o `sub_mb_type[mbPartIdx]` es igual a `B_Direct_8x8`, se llama al proceso de cálculo de vectores de movimiento `luma` para sectores `B_Skip`, `B_Direct_16x16` y `B_Direct_8x8` descrito en la subcláusula 8.4.1.2, pasándole como argumentos `mbPartIdx` y `subMbPartIdx`, y se obtiene como resultado los vectores de movimiento `mvL0`, `mvL1`, los índices de referencia `refIdxL0`, `refIdxL1`, el conteo de vectores de movimiento de subpartición `subMvCnt` y los indicadores de utilización de predicción, `predFlagL0` y `predFlagL1`.
- De lo contrario, siendo `X` igual a 0 ó 1 en las variables `predFlagLX`, `mvLX`, `refIdxLX` y `Pred_LX` y en los elementos sintácticos `ref_idx_IX` y `mvd_IX`, se aplica lo siguiente.



- Las variables refIdxLX y predFlagLX se calculan como sigue.
  - Si MbPartPredMode( mb\_type, mbPartIdx ) o SubMbPredMode( sub\_mb\_type[ mbPartIdx ] ) es igual a Pred\_LX o BiPred,
 
$$\text{refIdxLX} = \text{ref\_idx\_IX}[ \text{mbPartIdx} ] \quad (8-120)$$

$$\text{predFlagLX} = 1 \quad (8-121)$$
  - De lo contrario, las variables refIdxLX y predFlagLX se especifican así:
 
$$\text{refIdxLX} = -1 \quad (8-122)$$

$$\text{predFlagLX} = 0 \quad (8-123)$$
- La variable subMvCnt que lleva el conteo del vector de movimiento de subpartición se fija igual a predFlagL0 + predFlagL1.
- Si predFlagLX es igual a 1, se llama al proceso de cálculo de predicción de vectores de movimiento luma descrito en la subcláusula 8.4.1.3, pasándole como argumentos mbPartIdx, subMbPartIdx, refIdxLX y el sufijo de lista LX, y el resultado es mvplX. Los vectores de movimiento luma se calculan mediante las siguientes expresiones:
 
$$\text{mvLX}[ 0 ] = \text{mvplX}[ 0 ] + \text{mvd\_IX}[ \text{mbPartIdx} ][ \text{subMbPartIdx} ][ 0 ] \quad (8-124)$$

$$\text{mvLX}[ 1 ] = \text{mvplX}[ 1 ] + \text{mvd\_IX}[ \text{mbPartIdx} ][ \text{subMbPartIdx} ][ 1 ] \quad (8-125)$$

Para calcular las variables de los vectores de movimiento croma se aplica lo siguiente. Cuando predFlagLX (siendo X 0 ó 1 ) es igual a 1, se llama al proceso de cálculo de vectores de movimiento croma descrito en la subcláusula 8.4.1.4, pasándole como argumentos mvLX y refIdxLX y el resultado es mvCLX.

#### 8.4.1.1 Proceso de cálculo de vectores de movimiento luma para macrobloques obviados en vectores P y SP

Se llama a este proceso cuando mb\_type es igual a P\_Skip.

Este proceso genera como resultado el vector de movimiento mvL0 y el índice de referencia refIdxL0.

El índice de referencia refIdxL0 de un macrobloque obviado es:

$$\text{refIdxL0} = 0 \quad (8-126)$$

Para calcular el vector de movimiento mvL0 de un macrobloque de tipo P\_Skip se aplica a lo siguiente:

- Fíjese currSubMbType igual a sub\_mb\_type[ 0 ]. Se llama al proceso especificado en la subcláusula 8.4.1.3.2 pasándole como argumentos mbPartIdx puesto a 0, subMbPartIdx puesto a 0, currSubMbType, y el sufijo de lista L0, y el resultado se asigna a mbAddrA, mbAddrB, mvL0A, mvL0B, refIdxL0A, y refIdxL0B.
- La variable mvL0 se calcula del modo siguiente:
  - Si se cumple alguna de las condiciones siguientes, las dos componentes del vector en movimiento mvL0 se ponen a 0:
    - mbAddrA no está disponible
    - mbAddrB no está disponible
    - refIdxL0A es igual a 0 y las dos componentes de mvL0A son iguales a 0
    - refIdxL0B es igual a 0 y las dos componentes de mvL0B son iguales a 0
  - De lo contrario, se llama al proceso de cálculo de predicción de vectores de movimiento luma descrito en la subcláusula 8.4.1.3 pasándole como argumentos mbPartIdx = 0, subMbPartIdx = 0, refIdxL0, y el sufijo de lista L0 y el resultado se asigna a mvL0.

NOTA – El resultado se asigna directamente a mvL0 dado que el predictor es igual al vector de movimiento real.

#### 8.4.1.2 Proceso de cálculo de vector de movimiento luma para sectores B\_Skip, B\_Direct\_16x16 y B\_Direct\_8x8

Se llama a este proceso cuando mb\_type es igual a B\_Skip o B\_Direct\_16x16, o sub\_mb\_type[ mbPartIdx ] es igual a B\_Direct\_8x8.

Este proceso acepta como argumentos mbPartIdx y subMbPartIdx.

Este proceso genera como resultado los índices de referencia refIdxL0, refIdxL1, los vectores de movimiento mvL0 y mvL1, el conteo de vectores de movimiento de subpartición subMvCnt, y los indicadores de utilización de lista de predicción, predFlagL0 y predFlagL1.

El proceso de cálculo depende del valor de direct\_spatial\_mv\_pred\_flag, que aparece en la sintaxis de cabecera de sector en el tren de bits, según se especifica en la subcláusula 7.3.3, y se define del modo siguiente:

- Si direct\_spatial\_mv\_pred\_flag es igual a 1, el modo con que se calculan los resultados de este proceso se denomina modo de predicción directa espacial.
- De lo contrario (direct\_spatial\_mv\_pred\_flag es igual a 0), el modo con que se calculan los resultados de este proceso se denomina modo de predicción directa temporal.

Los dos modos de predicción directa, espacial y temporal, utilizan los vectores de movimiento coubicados y los índices de referencia como se especifica en la subcláusula 8.4.1.2.1.

Los vectores de movimiento y los índices de referencia se calculan del modo siguiente:

- Si se utiliza el modo de predicción directa espacial, se utiliza el modo de predicción directo de vectores de movimiento y de índices de referencia especificado en la subcláusula 8.4.1.2.2, donde subMvCnt es uno de los resultados.
- De lo contrario (se utiliza el modo de predicción directa temporal), se utiliza el modo de predicción directa de vectores de movimiento y de índices de referencia especificado en la subcláusula 8.4.1.2.3, y la variable subMvCnt se calcula así.
  - Si subMbPartIdx es igual a 0, subMvCnt se fija en 2.
  - De lo contrario (subMbPartIdx no es igual a 0), subMvCnt se fija igual a 0.

##### 8.4.1.2.1 Proceso de cálculo de particiones submacrobloque 4x4 coubicadas

Este proceso acepta como argumentos mbPartIdx y subMbPartIdx.

El proceso genera como resultado la imagen colPic, el macrobloque coubicado mbAddrCol, el vector de movimiento mvCol, el índice de referencia refIdxCol y la variable vertMvScale (cuyos valores posibles son One\_To\_One, Frm\_To\_Fld o Fld\_To\_Frm).

Sea firstRefPicL1 la imagen de referencia a la que apunta RefPicList1[ 0 ].

Si firstRefPicL1 es un cuadro o un par de campos complementarios, sean firstRefPicL1Top y firstRefPicL1Bottom los campos superior e inferior de firstRefPicL1, y sean las dos variables definidas así:

$$\text{topAbsDiffPOC} = \text{Abs}(\text{DiffPicOrderCnt}(\text{firstRefPicL1Top}, \text{CurrPic})) \quad (8-127)$$

$$\text{bottomAbsDiffPOC} = \text{Abs}(\text{DiffPicOrderCnt}(\text{firstRefPicL1Bottom}, \text{CurrPic})) \quad (8-128)$$

La variable colPic especifica la imagen que contiene el macrobloque coubicado, según se especifica en el cuadro 8-5.

**Cuadro 8-5 – Especificación de la variable colPic**

field_pic_flag	El primer elemento de RefPicList1 es ...	mb_field_decoding_flag	condición adicional	colPic
1	Un campo de un cuadro decodificado			El cuadro que contiene firstRefPicL1
	Un campo decodificado			firstRefPicL1
0	Un cuadro decodificado			firstRefPicL1
	Un par de campos complementarios	0	topAbsDiffPOC < bottomAbsDiffPOC	El campo superior de firstRefPicL1
			topAbsDiffPOC >= bottomAbsDiffPOC	El campo inferior de firstRefPicL1
	Un par de campos complementarios	1	( CurrMbAddr & 1 ) == 0	El campo superior de firstRefPicL1
( CurrMbAddr & 1 ) != 0			El campo inferior de firstRefPicL1	

Cuando direct\_8x8\_inference\_flag es igual a 1, el valor de subMbPartIdx es:

$$\text{subMbPartIdx} = \text{mbPartIdx} \quad (8-129)$$

Sea PicCodingStruct( X ) una función, cuyo argumento X puede ser CurrPic o colPic. Esta función se especifica en el cuadro 8-6.

**Cuadro 8-6 – Especificación de PicCodingStruct( X )**

X se codifica con field_pic_flag igual a ...	mb_adaptive_frame_field_flag	PicCodingStruct( X )
1		FLD
0	0	FRM
0	1	AFRM

Se llama al proceso de barrido inverso de bloques luma 4x4 descrito en la subcláusula 6.4.3, pasándole como argumento luma4x4BlkIdx, siendo  $\text{luma4x4BlkIdx} = \text{mbPartIdx} * 4 + \text{subMbPartIdx}$ , y el resultado ( x, y ) se asigna a ( xCol, yCol ).

En el cuadro 8-7 se especifica la dirección del macrobloque cubicado mbAddrCol, yM, y la variable vertMvScale en dos pasos:

1. Especificación de una dirección de macrobloque mbAddrX en función de PicCodingStruct( CurrPic ) y PicCodingStruct( colPic ).

NOTA – Es imposible que los tipos de codificación de las imágenes CurrPic y colPic sean (FRM, AFRM) o (AFRM, FRM), dado que las imágenes de estos tipos deben estar separadas por una imagen IDR.

2. Especificación de mbAddrCol, yM, y vertMvScale en función de mb\_field\_decoding\_flag y la variable fieldDecodingFlagX, que se calcula del modo siguiente:

- Si el macrobloque mbAddrX en la imagen colPic es un macrobloque campo, fieldDecodingFlagX se pone a 1.
- De lo contrario (el macrobloque mbAddrX en la imagen colPic es un macrobloque cuadro), fieldDecodingFlagX se pone a 0.

Los valores no especificados en el cuadro 8-7 indican que el valor de la correspondiente variable no es pertinente para la fila del cuadro considerada.

$$\text{mbAddrCol1} = 2 * \text{PicWidthInMbs} * ( \text{CurrMbAddr} / \text{PicWidthInMbs} ) + ( \text{CurrMbAddr} \% \text{PicWidthInMbs} ) + \text{PicWidthInMbs} * ( \text{yCol} / 8 ) \quad (8-130)$$

$$\text{mbAddrCol2} = 2 * \text{CurrMbAddr} + ( \text{yCol} / 8 ) \quad (8-131)$$

$$\text{mbAddrCol3} = 2 * \text{CurrMbAddr} + \text{bottom\_field\_flag} \quad (8-132)$$

$$\text{mbAddrCol4} = \text{PicWidthInMbs} * ( \text{CurrMbAddr} / ( 2 * \text{PicWidthInMbs} ) ) + ( \text{CurrMbAddr} \% \text{PicWidthInMbs} ) \quad (8-133)$$

$$\text{mbAddrCol5} = \text{CurrMbAddr} / 2 \quad (8-134)$$

$$\text{mbAddrCol6} = 2 * ( \text{CurrMbAddr} / 2 ) + ( ( \text{topAbsDiffPOC} < \text{bottomAbsDiffPOC} ) ? 0 : 1 ) \quad (8-135)$$

$$\text{mbAddrCol7} = 2 * ( \text{CurrMbAddr} / 2 ) + ( \text{yCol} / 8 ) \quad (8-136)$$

**Cuadro 8-7 – Especificación de mbAddrCol, yM, y vertMvScale**

PicCodingStruct( CurrPic )	PicCodingStruct( colPic )	mbAddrX	mb_field_decoding_flag	fieldDecodingFlagX	mbAddrCol	yM	vertMvScale
FLD	FLD				CurrMbAddr	yCol	One_To_One
	FRM				mbAddrCol1	( 2 * yCol ) % 16	Frm_To_Fld
	AFRM	2*CurrMbAddr	0		mbAddrCol2	( 2 * yCol ) % 16	Frm_To_Fld
1				mbAddrCol3	yCol	One_To_One	
FRM	FLD				mbAddrCol4	8 * ( ( CurrMbAddr / PicWidthInMbs ) % 2 ) + 4 * ( yCol / 8 )	Fld_To_Frm
	FRM				CurrMbAddr	yCol	One_To_One
AFRM	FLD		0		mbAddrCol5	8 * ( CurrMbAddr % 2 ) + 4 * ( yCol / 8 )	Fld_To_Frm
			1		mbAddrCol5	yCol	One_To_One
	AFRM	CurrMbAddr	0		CurrMbAddr	yCol	One_To_One
			1		mbAddrCol6	8 * ( CurrMbAddr % 2 ) + 4 * ( yCol / 8 )	Fld_To_Frm
			0		mbAddrCol7	( 2 * yCol ) % 16	Frm_To_Fld
1		CurrMbAddr	yCol	One_To_One			

Sea mbPartIdxCol el índice de partición macrobloque de la partición coubicada, y sea subMbPartIdxCol el índice de partición macrobloque de la partición submacrobloque coubicada. Se asignará a mbPartIdxCol, la partición en el macrobloque mbAddrCol dentro de la imagen colPic que contiene la muestra ( xCol, yM ), y se asignará a subMbPartIdxCol la partición de submacrobloque dentro de la partición mbPartIdxCol que contenga la muestra ( xCol, yM ) en el macrobloque mbAddrCol dentro de la imagen colPic.

Los indicadores de utilización de predicción predFlagL0Col y predFlagL1Col se ponen, respectivamente, a PredFlagL0[ mbPartIdxCol ] y PredFlagL1[ mbPartIdxCol ], que son los indicadores de utilización de predicción que han sido asignados a la partición macrobloque mbAddrCol\mbPartIdxCol dentro de la imagen colPic.

El vector en movimiento mvCol y el índice de referencia refIdxCol se calculan del modo siguiente:

- Si el macrobloque mbAddrCol está codificado en el modo de predicción macrobloque Intra o los dos indicadores de utilización de predicción predFlagL0Col y predFlagL1Col son iguales a 0, el valor de las dos componentes de mvCol se pone a 0 y refIdxCol se pone a -1.
- De lo contrario, se aplica lo siguiente:
  - Si predFlagL0Col es igual a 1, el vector en movimiento mvCol y el índice de referencia refIdxCol se ponen a  $MvL0[mbPartIdxCol][subMbPartIdxCol]$  y  $RefIdxL0[mbPartIdxCol]$ , respectivamente, que son el vector de movimiento mvL0 y el índice de referencia refIdxL0 que han sido asignados a la partición (sub)macrobloque mbAddrCol\mbPartIdxCol\subMbPartIdxCol dentro de la imagen colPic.
  - De lo contrario (predFlagL0Col es igual a 0 y predFlagL1Col es igual a 1) el vector de movimiento mvCol y el índice de referencia refIdxCol se ponen a  $MvL1[mbPartIdxCol][subMbPartIdxCol]$  y  $RefIdxL1[mbPartIdxCol]$ , respectivamente, que son el vector de movimiento mvL1 y el índice de referencia refIdxL1 que han sido asignados a la partición (sub)macrobloque mbAddrCol\mbPartIdxCol\subMbPartIdxCol dentro de la imagen colPic.

#### 8.4.1.2.2 Proceso de cálculo del modo de predicción directa espacial de vectores en movimiento luma e índices de referencia

Se llama a este proceso cuando direct\_spatial\_mv\_pred\_flag es igual a 1 y se cumple alguna de las siguientes condiciones:

- mb\_type es igual a B\_Skip;
- mb\_type es igual a B\_Direct\_16x16;
- sub\_mb\_type[ mbPartIdx ] es igual a B\_Direct\_8x8.

Este proceso acepta como argumentos mbPartIdx, subMbPartIdx.

Este proceso genera como resultados los índices de referencia refIdxL0, refIdxL1, los vectores de movimiento mvL0 y mvL1, el contador de vectores de movimiento de subpartición subMvCnt y los indicadores de utilización de lista de predicción predFlagL0 y predFlagL1.

Los índices de referencia refIdxL0 y refIdxL1 y la variable directZeroPredictionFlag se calculan mediante los siguientes pasos:

1. Fíjese la variable currSubMbType igual a sub\_mb\_type[ mbPartIdx ].
2. Se llama al proceso definido en la subcláusula 8.4.1.3.2 pasándole como argumentos mbPartIdx = 0, subMbPartIdx = 0, currSubMbType y el sufijo de lista L0, y el resultado se asigna a los vectores de movimiento mvL0N y a los índices de referencia refIdxL0N, siendo N-A, B o C.
3. Se llama al proceso descrito en la subcláusula 8.4.1.3.2 pasándolo como argumentos mbPartIdx = 0, subMbPartIdx = 0, currSubMbType y el sufijo de lista L1 y el resultado se asigna al vector de movimiento mvL1N y a los índices de referencia refIdxL1N, siendo N-A, B o C.

NOTA – Los vectores de movimiento mvL0N, mvL1N y los índices de referencia refIdxL0N y refIdxL1N son idénticos para todas las particiones submacrobloques 4x4 de un macrobloque.

4. Los índices de referencia refIdxL0, refIdxL1 y directZeroPredictionFlag se calculan mediante las siguientes expresiones:

$$\text{refIdxL0} = \text{MinPositive}(\text{refIdxL0A}, \text{MinPositive}(\text{refIdxL0B}, \text{refIdxL0C})) \quad (8-137)$$

$$\text{refIdxL1} = \text{MinPositive}(\text{refIdxL1A}, \text{MinPositive}(\text{refIdxL1B}, \text{refIdxL1C})) \quad (8-138)$$

$$\text{directZeroPredictionFlag} = 0 \quad (8-139)$$

donde:

$$\text{MinPositive}(x, y) = \begin{cases} \text{Min}(x, y) & \text{si } x \geq 0 \text{ e } y \geq 0 \\ \text{Max}(x, y) & \text{en otro caso} \end{cases} \quad (8-140)$$

5. Cuando los dos índices refIdxL0 y refIdxL1 son menores que 0:

$$\text{refIdxL0} = 0 \quad (8-141)$$

$$\text{refIdxL1} = 0 \quad (8-142)$$

$$\text{directZeroPredictionFlag} = 1 \quad (8-143)$$

Se llama al proceso descrito en la subcláusula 8.4.1.2.1 pasándole como argumentos mbPartIdx y subMbPartIdx y el resultado se asigna a refIdxCol y mvCol.

La variable colZeroFlag se calcula del modo siguiente:

- Si se cumplen todas las condiciones siguientes, colZeroFlag se pone a 1:
  - la imagen de referencia a la que hace referencia RefPicList1[0] es una imagen de referencia de corto alcance;
  - refIdxCol es igual a 0;
  - las dos componentes del vector de movimiento mvCol[ 0 ] y mvCol[ 1 ] están entre –1 y 1, medido en las siguientes unidades:
    - si el macrobloque coubicado es un macrobloque de cuadro, las unidades de mvCol[ 0 ] y mvCol[ 1 ] son unidades de un cuarto de muestra cuadro luma;
    - de lo contrario (el macrobloque coubicado es un macrobloque de campo), las unidades de mvCol[ 0 ] y mvCol[ 1 ] son unidades de un cuarto de mostrar campo luma.

NOTA – Para determinar la condición anterior, no se cambia de escala el valor de mvCol[ 1 ] a las unidades de un vector de movimiento del macrobloque considerado en los casos en que el macrobloque considerado es un macrobloque cuadro y el macrobloque coubicado es un macrobloque campo, o cuando el macrobloque considerado es un macrobloque campo y el macrobloque coubicado es un macrobloque cuadro. Este aspecto difiere de la utilización mvCol[ 1 ] en el modo directo temporal especificado en la subcláusula 8.4.1.2.3, en el cual se aplica un cambio de escala al vector de movimiento del macrobloque coubicado con objeto de utilizar las mismas unidades que las del vector de movimiento en el macrobloque considerado, en cuyo caso se utilizan las ecuaciones 8-146 u 8-147.

- De lo contrario, colZeroFlag se pone a 0.

Los vectores de movimiento mvLX (siendo X 0 ó 1) se calculan del modo siguiente:

- Si se cumple alguna de las condiciones siguientes, las dos componentes del vector de movimiento mvLX se ponen a 0:
  - directZeroPredictionFlag es igual a 1;
  - refIdxLX es menor que 0;
  - refIdxLX es igual a 0 y colZeroFlag es igual a 1.
- De lo contrario, se llama al proceso descrito en la subcláusula 8.4.1.3 pasándole como argumentos mbPartIdx = 0, subMbPartIdx = 0, refIdxLX, y el sufijo de lista LX, y el resultado se asigna a mvLX.

NOTA – En este último caso, el vector de movimiento que devuelve el proceso mvLX es idéntico para todas las particiones submacrobloque 4x4 del macrobloque.

Los indicadores de utilización de predicción predFlagL0 y predFlagL1 se calculan de acuerdo con el cuadro 8-8.

**Cuadro 8-8 – Asignación de los indicadores de utilización de predicción**

<b>refIdxL0</b>	<b>refIdxL1</b>	<b>predFlagL0</b>	<b>predFlagL1</b>
>= 0	>= 0	1	1
>= 0	< 0	1	0
< 0	>= 0	0	1

La variable subMvCnt se calcula así:

- Si subMbPartIdx es diferente de 0 o direct\_8x8\_inference\_flag es igual a 0, subMvCnt se fija igual a 0.
- De lo contrario (subMbPartIdx es igual a 0 y direct\_8x8\_inference\_flag es igual a 1), subMvCnt se fija igual a predFlagL0 + predFlagL1.

#### **8.4.1.2.3 Proceso de cálculo del modo de predicción directa temporal de vectores de movimiento luma e índices de referencia**

Se llama a este proceso cuando direct\_spatial\_mv\_pred\_flag es igual a 0 y se cumple alguna de las siguientes condiciones:

- mb\_type es igual a B\_Skip;
- mb\_type es igual a B\_Direct\_16x16;
- sub\_mb\_type[ mbPartIdx ] es igual a B\_Direct\_8x8.

Este proceso acepta como argumentos mbPartIdx y subMbPartIdx.

Este proceso genera como resultado los vectores de movimiento mvL0 y mvL1, los índices de referencia refIdxL0 y refIdxL1 y los indicadores de utilización de lista de predicción, predFlagL0 y predFlagL1.

Se llama al proceso descrito en la subcláusula 8.4.1.2.1 pasándole como argumentos mbPartIdx, subMbPartIdx y el resultado se asigna a colPic, mbAddrCol, mvCol, refIdxCol, y vertMvScale.

Los índices de referencia refIdxL0 y refIdxL1 se calculan del modo siguiente:

$$\text{refIdxL0} = ( (\text{refIdxCol} < 0) ? 0 : \text{MapColToList0}(\text{refIdxCol}) ) \quad (8-144)$$

$$\text{refIdxL1} = 0 \quad (8-145)$$

NOTA – Si el macrobloque considerado es un macrobloque campo, refIdxL0 y refIdxL1 son los índices de una lista de campos; de lo contrario (el macrobloque considerado es un macrobloque cuadro), refIdxL0 y refIdxL1 son los índices de una lista de cuadros o de pares de campos de referencia complementarios.

Sea refPicCol un cuadro, un campo o un par de campos complementarios al que apuntaba al índice de referencia refIdxCol cuando se decodificó el macrobloque coubicado mbAddrCol dentro de la imagen colPic. La función MapColToList0( refIdxCol ) se define del modo siguiente:

- Si vertMvScale es igual a One\_To\_One, se cumple lo siguiente:
  - Si field\_pic\_flag es igual a 0 y el macrobloque actual es un macrobloque de campo, se aplica lo siguiente:
    - Defínase refIdxL0Frm como el índice de referencia de menor valor en la lista de imágenes de referencia considerada RefPicList0 que apunta al cuadro o al par de campos complementarios que contienen el campo refPicCol. RefPicList0 contendrá una variable PicNum o LongTermPicNum que apunta al cuadro o al par de campos complementarios que contienen refPicCol. El valor resultante de MapColToList0( ) es el siguiente.

- Si el campo al que hace referencia refIdxCol tiene la misma paridad que el macrobloque considerado, MapColToList0( refIdxCol ) da como resultado el índice de referencia ( refIdxL0Frm << 1 ).
  - De lo contrario (el campo al que hace referencia refIdxCol tiene la paridad opuesta a la del macrobloque considerado), MapColToList0( refIdxCol ) da como resultado el índice de referencia ( ( refIdxL0Frm << 1 ) + 1 ).
  - De lo contrario (field\_pic\_flag es igual a 1 o el macrobloque considerado es un macrobloque de cuadro), MapColToList0( refIdxCol ) devuelve el índice de referencia de menor valor refIdxL0 de la lista de imágenes de referencia considerada RefPicList0 que apunta a refPicCol. RefPicList0 contendrá la variable PicNum o LongTermPicNum que hace referencia a refPicCol.
  - De lo contrario, si vertMvScale es igual a Frm\_To\_Fld, se aplica lo siguiente:
    - Si \_pic\_flag es igual a 0, defínase refIdxL0Frm como el índice de referencia de menor valor en la lista de índices de referencia considerada RefPicList0 que apunta a refPicCol. MapColToList0( refIdxCol ) da como resultado el índice de referencia ( refIdxL0Frm << 1 ). RefPicList0 contendrá una variable PicNum o LongTermPicNum que apunta a refPicCol.
    - De lo contrario (field\_pic\_flag es igual a 1), MapColToList0( refIdxCol ) devuelve el índice de referencia de menor valor refIdxL0 de la lista de imágenes de referencia considerada RefPicList0 que apunta al campo de refPicCol con la misma paridad que la imagen considerada CurrPic. RefPicList0 contendrá la variable PicNum o LongTermPicNum que apunta al campo de refPicCol con la misma paridad que la imagen considerada CurrPic.
  - De lo contrario (vertMvScale es igual a Fld\_To\_Frm), MapColToList0( refIdxCol ) devuelve el índice de referencia de menor valor refIdxL0 de la lista de imágenes de referencia considerada RefPicList0 que apunta al cuadro o par de campos complementarios que contiene refPicCol. RefPicList0 contendrá la variable PicNum o LongTermPicNum que apunta al campo o par de campos complementarios que contiene refPicCol.
- NOTA – Es posible marcar como "utilizada para la referencia de largo alcance" la imagen de referencia decodificada que se había marcado como "utilizada como referencia de corto alcance", cuando hizo referencia a ésta en el proceso de decodificación de la imagen que contiene el macrobloque coubicado, antes de utilizarla como referencia para el modo de predicción Inter directa del macrobloque considerado.

La componente vertical de mvCol se modifica en función del valor de vertMvScale.

- Si vertMvScale es igual a Frm\_To\_Fld
 
$$mvCol[ 1 ] = mvCol[ 1 ] / 2 \quad (8-146)$$
- De lo contrario, si vertMvScale es igual a Fld\_To\_Frm
 
$$mvCol[ 1 ] = mvCol[ 1 ] * 2 \quad (8-147)$$
- De lo contrario (vertMvScale es igual a One\_To\_One), no se modifica mvCol[ 1 ].

Las variables currPicOrField, pic0, y pic1, se calculan como sigue.

- Si field\_pic\_flag es igual a 0 y el macrobloque considerado es un macrobloque de campo, se aplica lo siguiente:
  - currPicOrField es el campo de la imagen considerada CurrPic que tiene la misma paridad que el macrobloque considerado.
  - pic1 es el campo de RefPicList1[0] que tiene la misma paridad que el macrobloque considerado.



- Definase frame0 como el cuadro o par de campos complementarios a los que hace referencia RefPicList0[ refIdxL0 / 2 ].
- La variable pic0 se calcula como sigue:
  - Si refIdxL0 % 2 es igual a 0, pic0 es el campo de frame0 que tiene la misma paridad que el macrobloque actual.
  - De lo contrario (refIdxL0 % 2 no es igual a 0), pic0 es el campo de frame0 con paridad opuesta a la del macrobloque considerado.
- De lo contrario (field\_pic\_flag es igual 1 o el macrobloque considerado es un macrobloque de cuadro), currPicOrField es la imagen considerada CurrPic, pic1 es la imagen de referencia decodificada a la que apunta RefPicList1[ 0 ], y pic0 es la imagen de referencia decodificada a la que apunta RefPicList0[ refIdxL0 ].

Los dos vectores de movimiento mvL0 y mvL1 de cada partición submacrobloque 4x4 del macrobloque considerado se calcula del modo siguiente:

NOTA – A menudo sucede que muchas particiones submacrobloque 4x4 comparten los mismos vectores de movimiento y las mismas imágenes de referencia. En esos casos, es posible utilizar la compensación de movimiento en modo directo temporal para calcular los valores de muestras de predicción Inter en unidades más grandes que bloques 4x4 de muestras luma. Por ejemplo, cuando direct\_8x8\_inference\_flag es igual a 1, al menos cada cuadrante de muestras luma 8x8 del macrobloque comparte los mismos vectores de movimiento y las mismas imágenes de referencia.

- Si el índice de referencia refIdxL0 apunta a una imagen de largo alcance, o DiffPicOrderCnt( pic1, pic0 ) es igual a 0, , los vectores de movimiento mvL0, mvL1 para la partición de modo directo son:

$$mvL0 = mvCol \quad (8-148)$$

$$mvL1 = 0 \quad (8-149)$$

- De lo contrario, los vectores de movimiento mvL0, mvL1 se calculan como una versión a escala del vector de movimiento mvCol de la partición submacrobloque coubicada, según se especifica a continuación (véase figura 8-2).

$$tx = ( 16384 + Abs( td / 2 ) ) / td \quad (8-150)$$

$$DistScaleFactor = Clip3( -1024, 1023, ( tb * tx + 32 ) >> 6 ) \quad (8-151)$$

$$mvL0 = ( DistScaleFactor * mvCol + 128 ) >> 8 \quad (8-152)$$

$$mvL1 = mvL0 - mvCol \quad (8-153)$$

donde tb y td se calculan de la siguiente manera:

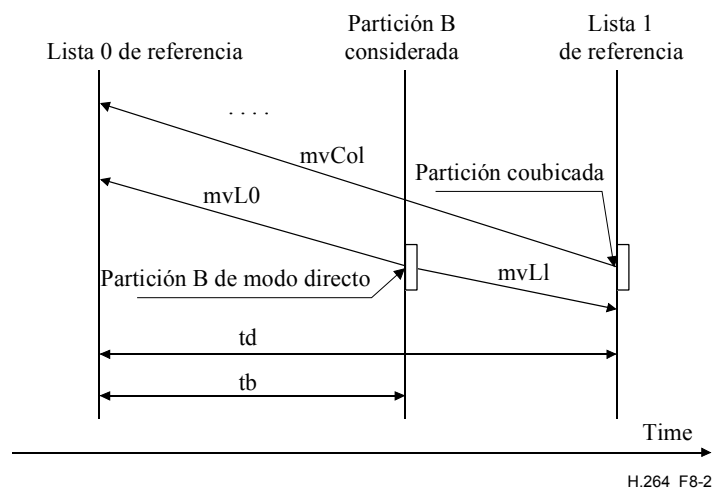
$$tb = Clip3( -128, 127, DiffPicOrderCnt( currPicOrField, pic0 ) ) \quad (8-154)$$

$$td = Clip3( -128, 127, DiffPicOrderCnt( pic1, pic0 ) ) \quad (8-155)$$

NOTA – mvL0 y mvL1 no pueden sobrepasar los valores especificados en el anexo A.

Los dos indicadores de utilización de predicción predFlagL0 y predFlagL1 se ponen a 1.

La figura 8-2 ilustra la obtención del vector de movimiento en modo directo temporal cuando la imagen considerada está temporalmente entre la lista 0 de imágenes de referencia y la lista 1 de imágenes de referencia.



**Figura 8-2 – Obtención de vector de movimiento en modo directo temporal (informativo)**

### 8.4.1.3 Proceso de cálculo de predicción de vectores de movimiento luma

Este proceso acepta como argumentos:

- el índice de la partición de macrobloque mbPartIdx;
- el índice de la partición de submacrobloque subMbPartIdx;
- el sufijo de lista LX;
- el índice de referencia de la partición considerada refIdxLX.

Este proceso genera como resultado la predicción mvpLX del vector de movimiento mvLX.

Fíjese currSubMbType igual a sub\_mb\_type [ mbPartIdx ].

Se llama al proceso de cálculo de bloques adyacente para datos de movimiento descrito en la subcláusula 8.4.1.3.2, al que se le pasan como argumentos mbPartIdx, subMbPartIdx, currSubMbType y el sufijo de lista LX, y se obtiene como resultados mbAddrN\mbPartIdxN\subMbPartIdxN, los índices de referencia refIdxLXN y los vectores de movimiento mvLXN, siendo N=A, B, o C.

Se llama al proceso de cálculo de predicción media del vector de movimiento luma descrito en la subcláusula 8.4.1.3.1, al que se le pasan como argumentos mbAddrN\mbPartIdxN\subMbPartIdxN, mvLXN, refIdxLXN, (siendo N=A, B, o C), y refIdxLX, y genera como resultado mvpLX, a no ser que se cumpla una de las condiciones siguientes.

- MbPartWidth( mb\_type ) es igual a 16, MbPartHeight( mb\_type ) es igual a 8, mbPartIdx es igual a 0, y refIdxLXB es igual a refIdxLX,

$$mvpLX = mvLXB \quad (8-156)$$

- MbPartWidth( mb\_type ) es igual a 16, MbPartHeight( mb\_type ) es igual a 8, mbPartIdx es igual a 1 y refIdxLXA es igual a refIdxLX,

$$mvpLX = mvLXA \quad (8-157)$$

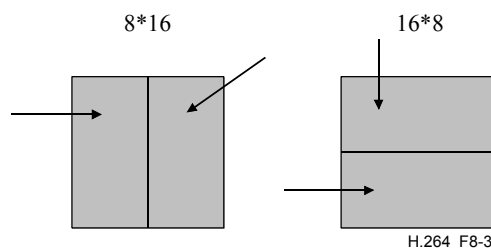
- MbPartWidth( mb\_type ) es igual a 8, MbPartHeight( mb\_type ) es igual a 16, mbPartIdx es igual a 0 y refIdxLXA es igual a refIdxLX,

$$mvpLX = mvLXA \quad (8-158)$$

- MbPartWidth( mb\_type ) es igual a 8, MbPartHeight( mb\_type ) es igual a 16, mbPartIdx es igual a 1 y refIdxLXC es igual a refIdxLX,

$$mvpLX = mvLXC \quad (8-159)$$

La figura 8-3 ilustra la predicción distinta de la predicción media descrita antes.



**Figura 8-3 – Predicción de segmentación direccional (informativo)**

#### 8.4.1.3.1 Proceso de cálculo de la predicción media de vectores de movimiento luma

Este proceso acepta como argumentos:

- las particiones adyacentes  $mbAddrN \setminus mbPartIdxN \setminus subMbPartIdxN$  (siendo  $N=A, B, o C$ );
- los vectores de movimiento  $mvLXN$  (siendo  $N=A, B, o C$ ) de las particiones adyacentes;
- los índices de referencia  $refIdxLXN$  (siendo  $N=A, B, o C$ ) de las particiones adyacentes; y
- el índice de referencia  $refIdxLX$  de la partición considerada.

Este proceso genera como resultado la predicción del vector de movimiento  $mvpLX$ .

La variable  $mvpLX$  se calcula del modo siguiente:

- Cuando ninguna de las dos particiones  $mbAddrB \setminus mbPartIdxB \setminus subMbPartIdxB$  y  $mbAddrC \setminus mbPartIdxC \setminus subMbPartIdxC$  está disponible y además  $mbAddrA \setminus mbPartIdxA \setminus subMbPartIdxA$  si está disponible,

$$mvLXB = mvLXA \quad (8-160)$$

$$mvLXC = mvLXA \quad (8-161)$$

$$refIdxLXB = refIdxLXA \quad (8-162)$$

$$refIdxLXC = refIdxLXA \quad (8-163)$$

- En función de los índices de referencia  $refIdxLXA$ ,  $refIdxLXB$ , o  $refIdxLXC$ , se aplica lo siguiente:

- Si uno y sólo uno de los índices de referencia  $refIdxLXA$ ,  $refIdxLXB$ , o  $refIdxLXC$  es igual al índice de referencia  $refIdxLX$  de la partición considerada, se aplica lo siguiente. Sea  $refIdxLXN$  el índice de referencia que es igual a  $refIdxLX$ , en ese caso el vector de movimiento  $mvLXN$  se asigna a la predicción del vector de movimiento  $mvpLX$ :

$$mvpLX = mvLXN \quad (8-164)$$

- De lo contrario, cada componente de la predicción del vector de movimiento  $mvpLX$  viene dado por la media de las correspondientes componentes del vector de movimiento  $mvLXA$ ,  $mvLXB$  y  $mvLXC$ :

$$mvpLX[0] = \text{Median}(mvLXA[0], mvLXB[0], mvLXC[0]) \quad (8-165)$$

$$mvpLX[1] = \text{Median}(mvLXA[1], mvLXB[1], mvLXC[1]) \quad (8-166)$$

#### 8.4.1.3.2 Proceso de cálculo de los datos de movimiento de particiones adyacentes

Este proceso acepta como argumentos:

- el índice de partición macrobloque  $mbPartIdx$ ;
- el índice de partición submacrobloque  $subMbPartIdx$ ;
- el tipo de sub-macrobloque considerado  $currSubMbType$ ;
- el sufijo de lista  $LX$ .

Este proceso genera como resultado (siendo N=A, B o C):

- $mbAddrN \setminus mbPartIdxN \setminus subMbPartIdxN$  que especifican las particiones adyacentes;
- los vectores de movimiento  $mvLXN$  de las particiones adyacentes; y
- los índices de referencia  $refIdxLXN$  de las particiones adyacentes.

Los pasos para calcular las particiones  $mbAddrN \setminus mbPartIdxN \setminus subMbPartIdxN$ , siendo N=A, B, o C, son los siguientes:

1. Sean  $mbAddrD \setminus mbPartIdxD \setminus subMbPartIdxD$  variables que especifican una partición adyacente adicional.
2. Se llama al proceso descrito en la subcláusula 6.4.7.5 pasándole como argumentos  $mbPartIdx$  y  $subMbPartIdx$ ,  $currSubMbType$  y el resultado se asigna a  $mbAddrN \setminus mbPartIdxN \setminus subMbPartIdxN$ , siendo N=A, B, C, o D.
3. Si la partición  $mbAddrC \setminus mbPartIdxC \setminus subMbPartIdxC$  no está disponible, se aplica lo siguiente:

$$mbAddrC = mbAddrD \quad (8-167)$$

$$mbPartIdxC = mbPartIdxD \quad (8-168)$$

$$subMbPartIdxC = subMbPartIdxD \quad (8-169)$$

Los vectores de movimiento  $mvLXN$  y los índices de referencia  $refIdxLXN$  (siendo N=A, B o C) se calculan del modo siguiente:

- Si la partición macrobloque o la partición submacrobloque  $mbAddrN \setminus mbPartIdxN \setminus subMbPartIdxN$  no está disponible o  $mbAddrN$  está codificada en modo de predicción intra o  $predFlagLX$  de  $mbAddrN \setminus mbPartIdxN \setminus subMbPartIdxN$  es igual a 0, las dos componentes de  $mvLXN$  se ponen a 0 y  $refIdxLXN$  se pone a -1.
- De lo contrario, se aplica lo siguiente:
  - Al vector de movimiento  $mvLXN$  y al índice de referencia  $refIdxLXN$  se les asigna, respectivamente,  $MvLX[ mbPartIdxN ][ subMbPartIdxN ]$  y  $RefIdxLX[ mbPartIdxN ]$ , que son el vector de movimiento  $mvLX$  y el índice de referencia  $refIdxLX$  que han sido asignados a la partición (sub)macrobloque  $mbAddrN \setminus mbPartIdxN \setminus subMbPartIdxN$ .
  - A continuación se calcula las variables  $mvLXN[ 1 ]$  y  $refIdxLXN$ :
    - Si el macrobloque considerado es un macrobloque campo y el macrobloque  $mbAddrN$  es un macrobloque cuadro
$$mvLXN[ 1 ] = mvLXN[ 1 ] / 2 \quad (8-170)$$
$$refIdxLXN = refIdxLXN * 2 \quad (8-171)$$
    - De lo contrario, si el macrobloque considerado es un macrobloque cuadro y el macrobloque  $mbAddrN$  es un macrobloque campo
$$mvLXN[ 1 ] = mvLXN[ 1 ] * 2 \quad (8-172)$$
$$refIdxLXN = refIdxLXN / 2 \quad (8-173)$$
    - De lo contrario, la componente vertical del vector de movimiento  $mvLXN[ 1 ]$  y el índice de referencia  $refIdxLXN$  no se modifican.

#### 8.4.1.4 Proceso de cálculo de vectores de movimiento croma

Este proceso acepta como argumentos un vector de movimiento luma  $mvLX$  y un índice de referencia  $refIdxLX$ .

Este proceso genera como resultado el vector de movimiento croma  $mvCLX$ .

Los vectores de movimiento croma se calculan a partir del vector de movimiento luma correspondiente. Dado que la precisión de los vectores de movimiento luma es de un cuarto de

muestra y que la resolución de los vectores croma es la mitad de la de los vectores luma, la precisión de los vectores de movimiento croma es de un octavo de muestra, es decir, el valor 1 para un vector de movimiento croma significa un desplazamiento de un octavo de muestra.

NOTA – Por ejemplo, cuando el vector luma se aplica a 8x16 muestras luma, el correspondiente vector croma se aplica a 4x8 muestras croma y, cuando el vector luma se aplica a 4x4 vectores luma, el correspondiente vector croma se aplica a 2x2 muestras croma.

Para calcular el vector de movimiento mvCLX se aplica lo siguiente:

- Si el macrobloque considerado es un macrobloque cuadro, las componentes horizontal y vertical del vector de movimiento croma mvCLX se calcula multiplicando por 2 las correspondientes componentes del vector de movimiento luma mvLX, de modo que las unidades de un cuarto de muestra mvLX se mapean en unidades de un octavo de muestra mvCLX.

$$mvCLX[0] = mvLX[0] \quad (8-174)$$

$$mvCLX[1] = mvLX[1] \quad (8-175)$$

- De lo contrario (el macrobloque considerado es un macrobloque campo), sólo se calcula la componente horizontal del vector de movimiento croma mvCLX[0] mediante la ecuación 8-174. La componente vertical del vector de movimiento croma mvCLX[1] es función de la paridad del campo considerado o del macrobloque considerado y de la imagen de referencia, a la cual apunta el índice de referencia refIdxLX. El valor de mvCLX[1] se obtiene a partir de mvLX[1] de acuerdo con el cuadro 8-9.

**Cuadro 8-9 – Cálculo de la componente vertical del vector croma en modo de codificación campo**

Condiciones de paridad		mvCLX[1]
Imagen de referencia (refIdxLX)	Campo considerado (imagen/macrobloque)	
Campo superior	Campo inferior	mvLX[1] + 2
Campo inferior	Campo superior	mvLX[1] - 2
En otro caso		mvLX[1]

#### 8.4.2 Proceso de decodificación de muestras de predicción inter

Este proceso acepta como argumentos:

- una partición de macrobloque mbPartIdx,
- una partición de submacrobloque subMbPartIdx,
- las variables que expresan la anchura y la altura de la partición, partWidth y partHeight,
- los vectores de movimiento luma mvL0 y mvL1 y los vectores de movimiento croma mvCL0 y mvCL1,
- los índices de referencia refIdxL0 y refIdxL1,
- los indicadores de utilización de lista de referencia, predFlagL0 y predFlagL1.

Este proceso genera como resultado

- las muestras de predicción Inter predPart, que son una matriz (partWidth)x(partHeight) predPart<sub>L</sub> de muestras luma de predicción y dos matrices (partWidth/2)x(partHeight/2) predPart<sub>Cb</sub>, predPart<sub>Cr</sub> de muestras croma de predicción, una para cada componente croma Cb y Cr.

Sean predPartL0<sub>L</sub> y predPartL1<sub>L</sub> dos matrices (partWidth)x(partHeight) de valores de muestras luma predichas y predPartL0<sub>Cb</sub>, predPartL1<sub>Cb</sub>, predPartL0<sub>Cr</sub>, y predPartL1<sub>Cr</sub> cuatro matrices (partWidth/2)x(partHeight/2) de valores de muestra croma predichas.

Sean las variables predFlagLX, RefPicListX, refIdxLX, refPicLX y predPartLX, siendo LX=L0 o L1.

Cuando predFlagLX es igual a 1, se aplica lo siguiente.

- El cuadro de referencia, que consta de una matriz bidimensional ordenada refPicLX<sub>L</sub> de muestras luma y dos matrices bidimensionales ordenadas refPicLX<sub>Cb</sub> y refPicLX<sub>Cr</sub> de muestras croma, se calcula mediante el proceso descrito en la subcláusula 8.4.2.1, al que se le pasa como argumentos refIdxLX y RefPicListX.
- Las matrices predPartLX<sub>L</sub>, predPartLX<sub>Cb</sub>, y predPartLX<sub>Cr</sub> se calculan mediante el proceso especificado en la subcláusula 8.4.2.2 al que se le pasan como argumentos la partición considerada especificada mediante mbPartIdx\subMbPartIdx, los vectores de movimiento mvLX, mvCLX, y las matrices de referencia refPicLX<sub>L</sub>, refPicLX<sub>Cb</sub>, y refPicLX<sub>Cr</sub>.

La matriz predPart<sub>C</sub> de muestras de predicción de la componente C, siendo C L, Cb, o Cr, se calcula mediante el proceso especificado en la subcláusula 8.4.2.3, al que se le pasan como argumentos y la partición considerada especificada mediante mbPartIdx y subMbPartIdx, las matrices predPartL0<sub>C</sub> y predPartL1<sub>C</sub>, y predFlagL0 y predFlagL1.

#### 8.4.2.1 Proceso de selección de imágenes de referencia

Este proceso acepta como argumentos el índice de referencia refIdxLX.

Este proceso genera como resultado una imagen de referencia, que es una matriz bidimensional de muestras luma refPicLX<sub>L</sub> y dos matrices bidimensionales de muestras croma refPicLX<sub>Cb</sub> y refPicLX<sub>Cr</sub>.

La lista de imágenes de referencia RefPicListX es una lista de variables PicNum (imágenes de referencia de corto alcance) y LongTermPicNum (imágenes de referencia de largo alcance) de cuadros de referencia previamente decodificados, pares de campos de referencia complementarios o campos de referencia desapareados que han sido marcados como "utilizados para referencia" según se especifica en la subcláusula 8.2.5.

El significado de PicNum y LongTermPicNum se especifica en función del field\_pic\_flag.

- Si field\_pic\_flag es igual a 1, todos los elementos de RefPicListX son variables PicNum y LongTermPicNum de campos de referencia decodificados o de campos de cuadros de referencia decodificados.
- De lo contrario (field\_pic\_flag es igual a 0), todos los elementos de RefPicListX son variables PicNum y LongTermPicNum de cuadros de referencia decodificados o pares de campos de referencia complementarios.

La lista de imágenes de referencia RefPicListX se calcula como se describe en la subcláusula 8.2.4.

Para calcular la imagen de referencia se aplica lo siguiente:

- Si field\_pic\_flag es igual a 1, el resultado será el campo de referencia o el campo de un cuadro de referencia al que apunta PicNum = RefPicListX[ refIdxLX ] o LongTermPicNum = RefPicListX[ refIdxLX ]. El campo de referencia o el campo de un cuadro de referencia resultante es una matriz (PicWidthInSamples<sub>L</sub>)x(PicHeightInSamples<sub>L</sub>) de muestras luma refPicLX<sub>L</sub> y dos matrices (PicWidthInSamples<sub>C</sub>)x(PicHeightInSamples<sub>C</sub>) de muestras croma refPicLX<sub>Cb</sub> y refPicLX<sub>Cr</sub>.
- De lo contrario (field\_pic\_flag es igual a 0), se aplica lo siguiente:
  - Si el macrobloque considerado es un macrobloque cuadro, el resultado será el cuadro de referencia o el par de campos de referencia complementarios al que apunta PicNum = RefPicListX[ refIdxLX ] o LongTermPicNum = RefPicListX[ refIdxLX ]. El cuadro de referencia o el par de campos de referencia complementarios resultantes

- constará de una matriz  $(\text{PicWidthInSamples}_L) \times (\text{PicHeightInSamples}_L)$  de muestras luma  $\text{refPicLX}_L$  y dos matrices  $(\text{PicWidthInSamples}_C) \times (\text{PicHeightInSamples}_C)$  de muestras croma  $\text{refPicLX}_{Cb}$  y  $\text{refPicLX}_{Cr}$ .
- De lo contrario (el macrobloque considerado es un macrobloque campo), se aplica lo siguiente:
    - Sea  $\text{refFrame}$  el cuadro de referencia o el par de campos de referencia complementarios al cual apunta  $\text{PicNum} = \text{RefPicListX}[\text{refIdxLX} / 2]$  o  $\text{LongTermPicNum} = \text{RefPicListX}[\text{refIdxLX} / 2]$ .
    - El campo de  $\text{refFrame}$  se selecciona del modo siguiente:
      - Si  $\text{refIdxLX} \% 2$  es igual a 0, el resultado será el campo de  $\text{refFrame}$  que tiene la misma paridad que el macrobloque considerado.
      - De lo contrario ( $\text{refIdxLX} \% 2$  es igual a 1), el resultado será el campo de  $\text{refFrame}$  que tiene la paridad contraria al macrobloque considerado.
    - El campo de referencia o el campo de un cuadro de referencia resultante estará formado por una matriz  $(\text{PicWidthInSamples}_L) \times (\text{PicHeightInSamples}_L / 2)$  de muestras luma  $\text{refPicLX}_L$  y dos matrices  $(\text{PicWidthInSamples}_C) \times (\text{PicHeightInSamples}_C / 2)$  de muestras croma  $\text{refPicLX}_{Cb}$  y  $\text{refPicLX}_{Cr}$ .

Las matrices de muestras de imágenes de referencia  $\text{refPicLX}_L$ ,  $\text{refPicLX}_{Cb}$ ,  $\text{refPicLX}_{Cr}$  corresponden a las matrices de muestras decodificadas  $S_L$ ,  $S_{Cb}$ ,  $S_{Cr}$  calculadas en la subcláusula 8.7 para imágenes previamente decodificadas.

#### 8.4.2.2 Proceso de interpolación de muestras fraccionario

Este proceso acepta como argumentos:

- la partición considerada descrita mediante su índice de partición  $\text{mbPartIdx}$  y su índice de partición submacrobloque  $\text{subMbPartIdx}$ ,
- la anchura y la altura de esta partición en unidades de muestra luma,  $\text{partWidth}$ ,  $\text{partHeight}$ ,
- un vector de movimiento luma  $\text{mvLX}$  en unidades de cuarto de muestra luma,
- un vector de movimiento croma  $\text{mvCLX}$  en unidades de octavo de muestra croma, y
- las matrices de muestras de imágenes de referencia seleccionadas  $\text{refPicLX}_L$ ,  $\text{refPicLX}_{Cb}$ , y  $\text{refPicLX}_{Cr}$ .

Este proceso genera como resultados:

- una matriz  $(\text{partWidth}) \times (\text{partHeight})$   $\text{predPartLX}_L$  de valores de muestra luma de predicción y
- dos matrices  $(\text{partWidth}/2) \times (\text{partHeight}/2)$   $\text{predPartLX}_{Cb}$ , y  $\text{predPartLX}_{Cr}$  de valores de muestra croma de predicción.

Sea  $(x_{A_L}, y_{A_L})$  la posición, en unidades de muestra entera, de la muestra luma superior izquierda de la partición considerada definida mediante  $\text{mbPartIdx} \setminus \text{subMbPartIdx}$  relativos a la posición de la muestra luma superior izquierda de la matriz bidimensional de muestras luma.

Sea  $(x_{\text{Int}_L}, y_{\text{Int}_L})$  la posición luma, en unidades de muestras enteras, y  $(x_{\text{Frac}_L}, y_{\text{Frac}_L})$  una traslación en unidades de cuartos de muestra. Estas variables sólo se utilizan en esta subcláusula para especificar las posiciones de muestras fraccionarias generales dentro de las matrices de muestras de referencia  $\text{refPicLX}_L$ ,  $\text{refPicLX}_{Cb}$ , y  $\text{refPicLX}_{Cr}$ .

Para cada posición de muestra luma  $(0 \leq x_L < \text{partWidth}, 0 \leq y_L < \text{partHeight})$  dentro de la matriz de muestras luma de predicción  $\text{predLX}_L$ , el correspondiente valor de la muestra luma predicha  $\text{predLX}_L[x_L, y_L]$  se calcula del modo siguiente:

$$xInt_L = xA_L + (mvLX[0] \gg 2) + x_L \quad (8-176)$$

$$yInt_L = yA_L + (mvLX[1] \gg 2) + y_L \quad (8-177)$$

$$xFrac_L = mvLX[0] \& 3 \quad (8-178)$$

$$yFrac_L = mvLX[1] \& 3 \quad (8-179)$$

- El valor de la muestra de predicción  $predLX_L[x_L, y_L]$  se obtiene mediante el proceso especificado en la subcláusula 8.4.2.2.1, al que se le pasan como argumentos  $(xInt_L, yInt_L)$ ,  $(xFrac_L, yFrac_L)$  y  $refPicLX_L$ .

Sea  $(xInt_C, yInt_C)$  una posición croma, en unidades de muestras enteras, y sea  $(xFrac_C, yFrac_C)$  una traslación en unidades de octavo de muestra. Estas variables se utilizan únicamente en esta subcláusula para especificar las posiciones de muestras fraccionarias generales dentro de las matrices de muestras de referencia  $refPicLX_{Cb}$  y  $refPicLX_{Cr}$ .

Para cada posición de muestras croma ( $0 \leq x_C < partWidth/2$ ,  $0 \leq y_C < partHeight/2$ ) dentro de las matrices de muestras croma de predicción  $predPartLX_{Cb}$  y  $predPartLX_{Cr}$ , los correspondientes valores de las muestras croma de predicción  $predPartLX_{Cb}[x_C, y_C]$  y  $predPartLX_{Cr}[x_C, y_C]$  se calculan del modo siguiente:

$$xInt_C = (xA_L \gg 1) + (mvCLX[0] \gg 3) + x_C \quad (8-180)$$

$$yInt_C = (yA_L \gg 1) + (mvCLX[1] \gg 3) + y_C \quad (8-181)$$

$$xFrac_C = mvCLX[0] \& 7 \quad (8-182)$$

$$yFrac_C = mvCLX[1] \& 7 \quad (8-183)$$

- El valor de la muestra de predicción  $predPartLX_{Cb}[x_C, y_C]$  se calcula mediante el proceso especificado en la subcláusula 8.4.2.2.2, al que se le pasan como argumentos  $(xInt_C, yInt_C)$ ,  $(xFrac_C, yFrac_C)$  y  $refPicLX_{Cb}$ .
- El valor de la muestra de predicción  $predPartLX_{Cr}[x_C, y_C]$  se calcula mediante el proceso especificado en la subcláusula 8.4.2.2.2, al que se le pasan como argumentos  $(xInt_C, yInt_C)$ ,  $(xFrac_C, yFrac_C)$  y  $refPicLX_{Cr}$ .

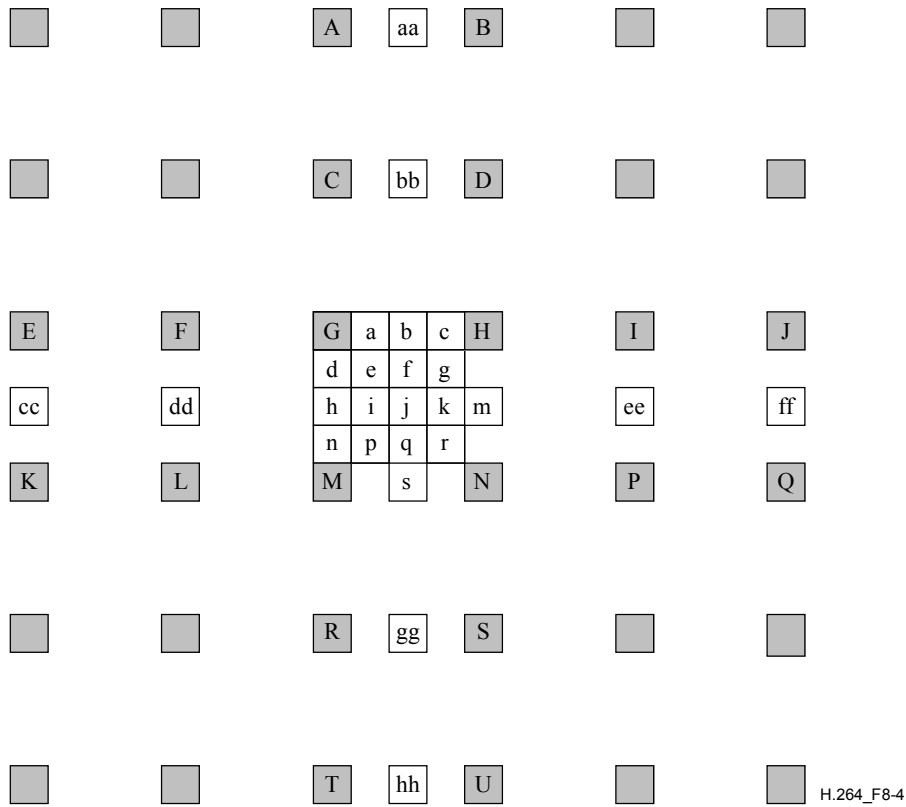
#### 8.4.2.2.1 Proceso de interpolación de muestras luma

Este proceso acepta como argumentos:

- la posición luma en unidades de muestra entera  $(xInt_L, yInt_L)$ ,
- una traslación de posición luma en unidades fraccionarias de muestra  $(xFrac_L, yFrac_L)$ , y
- la matriz de muestras luma de la imagen de referencia seleccionada  $refPicLX_L$

Este proceso genera como resultado el valor de la muestra luma predicha  $predPartLX_L[x_L, y_L]$ .





**Figura 8-4 – Posiciones de muestras enteras (bloques sombreados con letras mayúsculas) y posiciones de muestras fraccionarias (bloques sin sombrear y en letras minúsculas) para la interpolación luma de un cuarto de muestra**

La variable  $refPicHeightEffective_L$ , que es la altura de la matriz luma de la imagen de referencia efectiva, se calcula como sigue.

- Si  $MbaffFrameFlag$  es igual a 0 o  $mb\_field\_decoding\_flag$  es igual a 0,  $refPicHeightEffective_L$  se fija igual a  $PicHeightInSamples_L$ .
- De lo contrario ( $MbaffFrameFlag$  es igual a 1 y  $mb\_field\_decoding\_flag$  es igual a 1),  $refPicHeightEffective_L$  se fija igual a  $PicHeightInSamples_L / 2$ .

En la Figura 8-4, las posiciones indicadas con letras mayúsculas y en bloques sombreados representan muestras luma en posiciones de muestra entera dentro de la matriz bidimensional  $refPicLX_L$  de muestras luma. Estas muestras se pueden utilizar para obtener el valor de la muestra luma predicha  $predPartLX_L[x_L, y_L]$ . Las posiciones  $(xZ_L, yZ_L)$  de las correspondientes muestras luma Z, siendo  $Z=A, B, C, D, E, F, G, H, I, J, K, L, M, N, P, Q, R, S, T, o U$ , dentro de la matriz  $refPicLX_L$  de muestras luma se calcula del modo siguiente:

$$\begin{aligned} xZ_L &= Clip3(0, PicWidthInSamples_L - 1, xInt_L + xDZ_L) \\ yZ_L &= Clip3(0, refPicHeightEffective_L - 1, yInt_L + yDZ_L) \end{aligned} \quad (8-184)$$

El cuadro 8-10 especifica  $(xDZ_L, yDZ_L)$  para los diferentes valores de Z.

**Cuadro 8-10 – Posiciones luma de muestra entera diferenciales**

Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	P	Q	R	S	T	U
$xDZ_L$	0	1	0	1	-2	-1	0	1	2	3	-2	-1	0	1	2	3	0	1	0	1
$yDZ_L$	-2	-2	-1	-1	0	0	0	0	0	0	1	1	1	1	1	1	2	2	3	3

Dadas las muestras luma 'A' a 'U' en las posiciones de muestra entera (  $x_{A_L}, y_{A_L}$  ) a (  $x_{U_L}, y_{U_L}$  ), las muestras luma 'a' a 's' en las posiciones de muestras fraccionarias se calculan como se describe a continuación. Los valores de predicción luma en posiciones de media muestra se calcularán aplicando un filtro de 6 coeficientes cuyos valores son ( 1, -5, 20, 20, -5, 1 ). Los valores de predicción luma en las posiciones de un cuadro de muestras se calcularán haciendo la media de las muestras en las posiciones de muestra entera y de media muestra. A continuación se describe el proceso para cada posición fraccionaria.

- Para calcular las muestras en posiciones de media muestra indicadas con b; se calculará en primer lugar los valores intermedios indicados mediante  $b_1$ , para lo cual se aplicará un filtro de 6 coeficientes a las muestras de posición entera más cercanas, en sentido horizontal. Para calcular las muestras en las posiciones de media muestra indicadas mediante h, se calcularán en primer lugar los valores intermedios indicados como  $h_1$ , para lo cual se aplicará un filtro de 6 coeficientes a las muestras de posición entera más cercanas, en sentido vertical:

$$b_1 = ( E - 5 * F + 20 * G + 20 * H - 5 * I + J ) \quad (8-185)$$

$$h_1 = ( A - 5 * C + 20 * G + 20 * M - 5 * R + T ) \quad (8-186)$$

Los valores de predicción finales b y h se calculan mediante las siguientes expresiones:

$$b = \text{Clip1}( ( b_1 + 16 ) \gg 5 ) \quad (8-187)$$

$$h = \text{Clip1}( ( h_1 + 16 ) \gg 5 ) \quad (8-188)$$

- Para calcular las muestras en la posición de media muestra indicadas mediante j, se calculará en primer lugar el valor intermedio indicado como  $j_1$ , para lo cual se aplicará un filtro de 6 coeficientes a los valores intermedios de las posiciones de media muestra más cercanas en el sentido horizontal o vertical, dado que el resultado será el mismo.

$$j_1 = cc - 5 * dd + 20 * h_1 + 20 * m_1 - 5 * ee + ff, o \quad (8-189)$$

$$j_1 = aa - 5 * bb + 20 * b_1 + 20 * s_1 - 5 * gg + hh \quad (8-190)$$

donde los valores intermedios indicados como aa, bb, gg,  $s_1$  y hh se calculan aplicando un filtro de 6 coeficientes horizontalmente del mismo modo que en el cálculo  $b_1$ , y los valores intermedios indicados como cc, dd, ee,  $m_1$  y ff se calculan aplicando un filtro de 6 coeficientes en sentido vertical, del mismo modo que en el cálculo de  $h_1$ . El valor de predicción final j se calcula mediante la siguiente expresión:

$$j = \text{Clip1}( ( j_1 + 512 ) \gg 10 ) \quad (8-191)$$

- Los valores de predicciones finales de s y m se calculan a partir de  $s_1$  y  $m_1$  de igual manera que en el cálculo de b y h, es decir:

$$s = \text{Clip1}( ( s_1 + 16 ) \gg 5 ) \quad (8-192)$$

$$m = \text{Clip1}( ( m_1 + 16 ) \gg 5 ) \quad (8-193)$$

- Las muestras en las posiciones de un cuarto de muestra indicadas mediante a, c, d, n, f, i, k, y q se calculan haciendo la media, y redondeando por arriba, de las dos muestras más cercanas a las posiciones de muestra entera y media muestra mediante las siguientes expresiones:

$$a = ( G + b + 1 ) \gg 1 \quad (8-194)$$

$$c = ( H + b + 1 ) \gg 1 \quad (8-195)$$

$$d = ( G + h + 1 ) \gg 1 \quad (8-196)$$

$$n = ( M + h + 1 ) \gg 1 \quad (8-197)$$

$$f = ( b + j + 1 ) \gg 1 \quad (8-198)$$

$$i = ( h + j + 1 ) \gg 1 \quad (8-199)$$

$$k = ( j + m + 1 ) \gg 1 \quad (8-200)$$

$$q = ( j + s + 1 ) \gg 1 \quad (8-201)$$

- Las muestras en las posiciones de un cuarto de muestra indicadas mediante e, g, p, y r se calculan haciendo la media, y redondeando por arriba, de las dos muestras más cercanas a las posiciones de media muestra en dirección diagonal, mediante las siguientes expresiones:

$$e = (b + h + 1) \gg 1 \quad (8-202)$$

$$g = (b + m + 1) \gg 1 \quad (8-203)$$

$$p = (h + s + 1) \gg 1 \quad (8-204)$$

$$r = (m + s + 1) \gg 1 \quad (8-205)$$

La traslación de posición luma en unidades de muestra fraccionaria (  $xFrac_L$ ,  $yFrac_L$  ) indica cuál de las muestras luma generadas en las posiciones de muestra entera y de muestra fraccionaria se asigna al valor de la muestra luma predicha  $predPartLX_L[x_L, y_L]$ . La asignación se hace de conformidad con el cuadro 8-11. El resultado será el valor de  $predPartLX_L[x_L, y_L]$ .

**Cuadro 8-11 – Asignación de la muestra de predicción luma  $predPartLX_L[x_L, y_L]$**

$xFrac_L$	0	0	0	0	1	1	1	1	2	2	2	2	3	3	3	3
$yFrac_L$	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3
$predPartLX_L[x_L, y_L]$	G	d	h	n	a	e	i	p	b	f	j	q	c	g	k	r

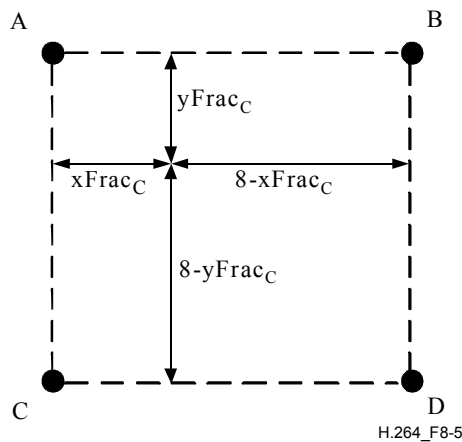
#### 8.4.2.2.2 Proceso de interpolación de muestras croma

Este proceso acepta como argumentos:

- una posición croma en unidades de muestra entera (  $xInt_C$ ,  $yInt_C$  ),
- una traslación de posición croma en unidades de muestra fraccionaria (  $xFrac_C$ ,  $yFrac_C$  ), y
- una muestra de componentes croma de la imagen de referencia seleccionada  $refPicLX_C$ .

Este proceso genera como resultado el valor de la muestra croma predicha  $predPartLX_C[x_C, y_C]$ .

En la figura 8-5 las posiciones indicadas mediante A, B, C, y D representan muestras croma en las posiciones de muestra entera dentro de la matriz bidimensional  $refPicLX_C$  de muestras croma.



**Figura 8-5 – Variables dependientes de posición de muestra fraccionaria en la interpolación croma y las muestras de posición entera de alrededor A, B, C y D**

La variable  $refPicHeightEffective_C$ , que es la altura de la matriz croma de la imagen de referencia efectiva, se calcula como sigue:

- Si  $MbaffFrameFlag$  es igual a 0 o  $mb\_field\_decoding\_flag$  es igual a 0,  $refPicHeightEffective_C$  se fija igual a  $PicHeightInSamples_C$ .
- De lo contrario ( $MbaffFrameFlag$  es igual a 1 y  $mb\_field\_decoding\_flag$  es igual a 1),  $refPicHeightEffective_C$  se fija igual a  $PicHeightInSamples_C / 2$ .

Las coordenadas de las muestras especificadas en las ecuaciones 8-206 a 8-213 se utilizan para generar el valor de la muestra croma predicha  $predPartLX_C[x_C, y_C]$ .

$$xA_C = \text{Clip3}(0, \text{PicWidthInSamples}_C - 1, xInt_C) \quad (8-206)$$

$$xB_C = \text{Clip3}(0, \text{PicWidthInSamples}_C - 1, xInt_C + 1) \quad (8-207)$$

$$xC_C = \text{Clip3}(0, \text{PicWidthInSamples}_C - 1, xInt_C) \quad (8-208)$$

$$xD_C = \text{Clip3}(0, \text{PicWidthInSamples}_C - 1, xInt_C + 1) \quad (8-209)$$

$$yA_C = \text{Clip3}(0, \text{refPicHeightEffective}_C - 1, yInt_C) \quad (8-210)$$

$$yB_C = \text{Clip3}(0, \text{refPicHeightEffective}_C - 1, yInt_C) \quad (8-211)$$

$$yC_C = \text{Clip3}(0, \text{refPicHeightEffective}_C - 1, yInt_C + 1) \quad (8-212)$$

$$yD_C = \text{Clip3}(0, \text{refPicHeightEffective}_C - 1, yInt_C + 1) \quad (8-213)$$

Dadas las muestras cromas A, B, C, y D en las posiciones de muestra entera especificadas en las ecuaciones 8-206 a 8-213, el valor de la muestra cromas predPartLX<sub>C</sub>[ x<sub>C</sub>, y<sub>C</sub> ] se calcula del modo siguiente:

$$\text{predPartLX}_C[x_C, y_C] = \left( \frac{(8 - xFrac_C) * (8 - yFrac_C) * A + xFrac_C * (8 - yFrac_C) * B + (8 - xFrac_C) * yFrac_C * C + xFrac_C * yFrac_C * D}{+ 32} \right) \gg 6 \quad (8-214)$$

### 8.4.2.3 Proceso de predicción de muestras ponderado

Este proceso acepta como argumentos:

- mbPartIdx: el índice de partición de la partición considerada
- subMbPartIdx: el índice de partición submacrobloque
- predFlagL0 y predFlagL1: los indicadores de utilización de listas de predicción
- predPartLX<sub>L</sub>: una matriz (partWidth)x(partHeight) de muestras luma de predicción (siendo LX L0 o L1, en función de predFlagL0 y predFlagL1)
- predPartLX<sub>Cb</sub> y predPartLX<sub>Cr</sub>: matrices (partWidth/2)x(partHeight/2) de muestras cromas de predicción, una para cada componente cromas, Cb y Cr (siendo LX= igual a L0 o L1, en función de predFlagL0 and predFlagL1)

Este proceso genera como resultados:

- predPart<sub>L</sub>: una matriz (partWidth)x(partHeight) de muestras luma de predicción y
- predPart<sub>Cb</sub>, y predPart<sub>Cr</sub>: matrices (partWidth/2)x(partHeight/2) de muestras cromas de predicción, una para cada componente cromas, Cb y Cr.

Para macrobloques o particiones con predFlagL0 igual a 1 en sectores P y SP, se aplica lo siguiente:

- Si weighted\_pred\_flag es igual a 0, se llama al proceso de predicción ponderada de muestras por defecto descrito en la subcláusula 8.4.2.3.1, al que se le pasan los mismos argumentos que al proceso descrito en esta subcláusula y que genera también los mismos resultados.
- De lo contrario (weighted\_pred\_flag es igual a 1) se llama al proceso de predicción ponderado explícito descrito en la subcláusula 8.4.2.3.2, al que se le pasan los mismos argumentos que el proceso descrito en esta subcláusula y cuyos resultados también son los mismos.

Para macrobloques o particiones con predFlagL0 o predFlagL1 igual a 1 en sectores B, se aplica lo siguiente.

- Si weighted\_bipred\_idc es igual a 0, se llama al proceso de predicción ponderada de muestras por defecto descrito en la subcláusula 8.4.2.3.1, al que se le pasa los mismos argumentos que al proceso descrito en esta subcláusula y que genera los mismos resultados.
- De lo contrario si weighted\_bipred\_idc es igual a 1, se llama al proceso de predicción ponderada de muestras explícito descrito en la subcláusula 8.4.2.3.2 para macrobloques o particiones con predFlagL0 o predFlagL1 igual a 1, al que se le pasan los mismos argumentos que al proceso descrito en esta subcláusula y se obtienen también los mismos resultados.

- De lo contrario (weighted\_bipred\_idc es igual a 2) se aplica a lo siguiente:
  - Si predFlagL0 es igual a 1 y predFlagL1 es igual a 1, se llama al proceso de predicción ponderada de muestras implícito descrito en la subcláusula 8.4.2.3.2, al cual se le pasan los mismos argumentos que al proceso descrito en esta subcláusula y que genera también los mismos resultados.
  - De lo contrario (predFlagL0 o predFlagL1 son igual a 1 pero no ambos) se llama al proceso de predicción ponderada de muestras por defecto descrito en la subcláusula 8.4.2.3.1, al cual se le pasan los mismos argumentos que al proceso descrito en esta subcláusula y que genera también los mismos resultados.

#### 8.4.2.3.1 Proceso de predicción ponderada de muestras por defecto

Este proceso acepta los mismos argumentos que el proceso especificado en la subcláusula 8.4.2.3.

Este proceso genera los mismos resultados que los especificados en la subcláusula 8.4.2.3.

En función de la componente para la que se calcule el bloque de predicción, se aplica lo siguiente:

- Si se calculan los valores de predicción de muestras luma  $\text{predPart}_L[x, y]$ , se aplica lo siguiente, siendo C igual a L, x igual a  $0 \dots \text{partWidth} - 1$ , e y igual a  $0 \dots \text{partHeight} - 1$ .
- De lo contrario, si se calculan los valores de predicción de muestras de la componente croma Cb  $\text{predPart}_{Cb}[x, y]$ , se aplica lo siguiente, siendo C igual a Cb, x igual a  $0 \dots \text{partWidth} / 2 - 1$  e y igual a  $0 \dots \text{partHeight} / 2 - 1$ .
- De lo contrario (se calcula los valores de predicción de muestras de la componente croma Cr  $\text{predPart}_{Cr}[x, y]$ ) se aplica lo siguiente, siendo C igual a Cr, x igual a  $0 \dots \text{partWidth} / 2 - 1$  e y igual a  $0 \dots \text{partHeight} / 2 - 1$ .

Los valores de muestras de predicción se calculan del modo siguiente:

- Si predFlagL0 es igual a 1 y predFlagL1 es igual a 0 para la partición considerada
 
$$\text{predPart}_c[x, y] = \text{predPartL0}_c[x, y] \quad (8-215)$$
- De lo contrario, si predFlagL0 es igual a 0 y predFlagL1 es igual a 1 para la partición considerada
 
$$\text{predPart}_c[x, y] = \text{predPartL1}_c[x, y] \quad (8-216)$$
- De lo contrario (predFlagL0 y predFlagL1 son iguales a 1 para la partición considerada),
 
$$\text{predPart}_c[x, y] = (\text{predPartL0}_c[x, y] + \text{predPartL1}_c[x, y] + 1) \gg 1 \quad (8-217)$$

#### 8.4.2.3.2 Proceso de predicción ponderada de muestras

Este proceso acepta los mismos argumentos que el proceso especificado en la subcláusula 8.4.2.3.

Este proceso genera los mismos argumentos que los especificados en la subcláusula 8.4.2.3.

En función de la componente para la que se calcule el bloque de predicción, se aplica lo siguiente:

- Si se calcula los valores de predicción de muestras luma  $\text{predPart}_L[x, y]$ , se aplica lo siguiente, siendo C igual a L, x igual a  $0 \dots \text{partWidth} - 1$ , e y igual a  $0 \dots \text{partHeight} - 1$ .
- De lo contrario, si se calculan los valores de predicción de muestras de la componente croma Cb  $\text{predPart}_{Cb}[x, y]$ , se aplica lo siguiente, siendo C igual a Cb, x igual a  $0 \dots \text{partWidth} / 2 - 1$ , e y igual a  $0 \dots \text{partHeight} / 2 - 1$ .
- De lo contrario (se calculan los valores de predicción de muestras de la componente croma Cr  $\text{predPart}_{Cr}[x, y]$ ) se aplica lo siguiente, siendo C igual a Cr, x igual a  $0 \dots \text{partWidth} / 2 - 1$ , e y igual a  $0 \dots \text{partHeight} / 2 - 1$ .

Los valores de muestras de predicción se calculan del modo siguiente:

- Si la partición tiene mbPartIdx\subMbPartIdx tiene predFlagL0 igual a 1 y predFlagL1 igual a 0, los valores de muestras de predicción finales predPartC[ x, y ] se calculan del modo siguiente:

$$\begin{aligned} & \text{if( logWD } \geq 1 ) \\ & \quad \text{predPartC[ x, y ]} = \text{Clip1}(( (\text{predPartL0C[ x, y ]} * w_0 + 2^{\text{logWD}-1}) \gg \text{logWD}) + o_0) \\ & \text{else} \\ & \quad \text{predPartC[ x, y ]} = \text{Clip1}(\text{predPartL0C[ x, y ]} * w_0 + o_0) \end{aligned} \quad (8-218)$$

- De lo contrario, si la partición en el mbPartIdx\subMbPartIdx tiene predFlagL0 igual a 0 y predFlagL1 igual a 1, los valores de las muestras predichas finales predPartC[ x, y ] se calculan del modo siguiente:

$$\begin{aligned} & \text{if( logWD } \geq 1 ) \\ & \quad \text{predPartC[ x, y ]} = \text{Clip1}(( (\text{predPartL1C[ x, y ]} * w_1 + 2^{\text{logWD}-1}) \gg \text{logWD}) + o_1) \\ & \text{else} \\ & \quad \text{predPartC[ x, y ]} = \text{Clip1}(\text{predPartL1C[ x, y ]} * w_1 + o_1) \end{aligned} \quad (8-219)$$

- De lo contrario (la partición mbPartIdx\subMbPartIdx tiene predFlagL0 y predFlagL1 iguales a 1), los valores de las muestras predichas finales predPartC[ x, y ] se calculan mediante la expresión

$$\text{predPartC[ x, y ]} = \text{Clip1}(( (\text{predPartL0C[ x, y ]} * w_0 + \text{predPartL1C[ x, y ]} * w_1 + 2^{\text{logWD}}) \gg (\text{logWD} + 1)) + ((o_0 + o_1 + 1) \gg 1)) \quad (8-220)$$

El cálculo de las variables que aparecen en las expresiones anteriores es el siguiente:

- Si weighted\_bipred\_idc es igual a 2 y slice\_type es igual a B, se utiliza la predicción ponderada de modo implícito de la siguiente manera:

$$\text{logWD} = 5 \quad (8-221)$$

$$o_0 = 0 \quad (8-222)$$

$$o_1 = 0 \quad (8-223)$$

y  $w_0$  y  $w_1$  se calculan del modo siguiente:

- Las variables currPicOrField, pic0 y pic1 se calculan de la siguiente manera:
  - Si field\_pic\_flag es igual a 0 y el macrobloque considerado es un macrobloque de campo, se aplica lo siguiente.
    - currPicOrField es el campo de la imagen considerada CurrPic que tiene la misma paridad que el macrobloque considerado.
    - Definase frame0 como el cuadro o el par de campos complementarios al que apunta RefPicList0[ refIdxL0 / 2 ].
  - La variable pic0 se calcula de la siguiente manera:
    - Si refIdxL0 % 2 es igual a 0, pic0 es el campo de frame0 que tiene la misma paridad que el macrobloque considerado.
    - De lo contrario (refIdxL0 % 2 no es igual a 0), pic0 es el campo del frame0 de paridad opuesta a la del macrobloque considerado.
  - Definase frame1 como el cuadro o par de campos complementarios al que apunta RefPicList1[ refIdxL1 / 2 ].
  - La variable pic1 se calcula de la siguiente manera:
    - Si refIdxL1 % 2 es igual a 0, pic1 es el campo de frame1 que tiene la misma paridad que el macrobloque considerado.
    - De lo contrario (refIdxL1 % 2 no es igual a 0), pic1 es el campo de frame1 con paridad opuesta a la del macrobloque considerado.

– De lo contrario (field\_pic\_flag es igual a 1 o el macrobloque considerado es un macrobloque de cuadro), currPicOrField es la imagen considerada CurrPic, pic1 es la imagen de referencia decodificada a la que apunta RefPicList1[ refIdxL1 ], y pic0 es la imagen de referencia codificada a la que apunta RefPicList0[ refIdxL0 ].

– Las variables tb, td, tx y DistScaleFactor se calculan a partir de los valores de currPicOrField, pic0, pic1 mediante las ecuaciones 8-154, 8-155, 8-150, y 8-151, respectivamente.

– Si DiffPicOrderCnt( pic1, pic0) es igual a 0, o pic1 o pic0 o ambas son una imagen de referencia de largo alcance o (DistScaleFactor >> 2) < -64 o (DistScaleFactor >> 2) > 128, w<sub>0</sub> y w<sub>1</sub> se calculan así:

$$w_0 = 32 \quad (8-224)$$

$$w_1 = 32 \quad (8-225)$$

– De lo contrario,

$$w_0 = 64 - (\text{DistScaleFactor} \gg 2) \quad (8-226)$$

$$w_1 = \text{DistScaleFactor} \gg 2 \quad (8-227)$$

– De lo contrario (weighted\_pred\_flag es igual a 1 en sectores P o SP o weighted\_bipred\_idc es igual a 1 en sectores B), se utiliza la predicción ponderada en modo explícito.

– Las variables refIdxL0WP y refIdxL1WP se calculan del modo siguiente:

– Si MbaffFrameFlag es igual a 1 y el macrobloque considerado es un macrobloque campo

$$\text{refIdxL0WP} = \text{refIdxL0} \gg 1 \quad (8-228)$$

$$\text{refIdxL1WP} = \text{refIdxL1} \gg 1 \quad (8-229)$$

– De lo contrario (MbaffFrameFlag es igual a 0 o el macrobloque considerado es un macrobloque cuadro)

$$\text{refIdxL0WP} = \text{refIdxL0} \quad (8-230)$$

$$\text{refIdxL1WP} = \text{refIdxL1} \quad (8-231)$$

– Las variables logWD, w<sub>0</sub>, w<sub>1</sub>, o<sub>0</sub>, y o<sub>1</sub> se calculan del modo siguiente:

– Si C en predPart<sub>C</sub>[ x, y ] se sustituye por L para muestras luma

$$\text{logWD} = \text{luma\_log2\_weight\_denom} \quad (8-232)$$

$$w_0 = \text{luma\_weight\_l0}[\text{refIdxL0WP}] \quad (8-233)$$

$$w_1 = \text{luma\_weight\_l1}[\text{refIdxL1WP}] \quad (8-234)$$

$$o_0 = \text{luma\_offset\_l0}[\text{refIdxL0WP}] \quad (8-235)$$

$$o_1 = \text{luma\_offset\_l1}[\text{refIdxL1WP}] \quad (8-236)$$

– De lo contrario (C en predPart<sub>C</sub>[ x, y ] se sustituye por Cb o Cr para muestras croma, siendo iCbCr = 0 para Cb, iCbCr = 1 para Cr),

$$\text{logWD} = \text{chroma\_log2\_weight\_denom} \quad (8-237)$$

$$w_0 = \text{chroma\_weight\_l0}[\text{refIdxL0WP}][iCbCr] \quad (8-238)$$

$$w_1 = \text{chroma\_weight\_l1}[\text{refIdxL1WP}][iCbCr] \quad (8-239)$$

$$o_0 = \text{chroma\_offset\_l0}[\text{refIdxL0WP}][iCbCr] \quad (8-240)$$

$$o_1 = \text{chroma\_offset\_l1}[\text{refIdxL1WP}][iCbCr] \quad (8-241)$$

Si se utiliza el modo de predicción ponderada de modo explícito la partición mbPartIdx\subMbPartIdx tiene predFlagL0 igual a 1 y predFlagL1 igual a 1, se deberá cumplir lo siguiente

$$-128 \leq w_0 + w_1 \leq ((\text{logWD} == 7) ? 127 : 128) \quad (8-242)$$

NOTA – Para la predicción ponderada de modo implícito, se garantiza que cada uno de los coeficientes de ponderación  $w_0$  y  $w_1$  están en la gama  $-64 \dots 128$  y siempre se cumplirá la restricción expresada en la ecuación 8-242, aunque ésta no se imponga de manera explícita. En el caso de predicción ponderada de modo explícito en la que  $\log_{2}WD$  es igual a 7, si alguno de los dos factores de ponderación  $w_0$  o  $w_1$  se calcula como igual a 128 (como consecuencia de `luma_weight_10_flag`, `luma_weight_11_flag`, `chroma_weight_10_flag`, o `chroma_weight_11_flag` igual a 0), el otro factor de ponderación ( $w_1$  o  $w_0$ ) debe tener un valor negativo con el fin de que se cumpla la ecuación 8-242 (y por lo tanto el otro indicador `luma_weight_10_flag`, `luma_weight_11_flag`, `chroma_weight_10_flag`, o `chroma_weight_11_flag` debe ser igual a 1).

## 8.5 Proceso de decodificación de coeficientes de transformada y proceso de reconstrucción de imágenes antes de aplicar el proceso de filtrado de bloques

Este proceso acepta como argumentos `Intra16x16DCLevel` (si lo hubiere), `Intra16x16ACLevel` (si lo hubiere), `LumaLevel` (si lo hubiere), `ChromaDCLevel`, `ChromaACLevel`, y las matrices de muestras de predicción  $\text{inter}$  o  $\text{intra}$  del macrobloque considerado para la componente que corresponda  $\text{pred}_L$ ,  $\text{pred}_{Cb}$ , o  $\text{pred}_{Cr}$ .

NOTA – Al decodificar un macrobloque en modo de predicción `Intra_4x4`, la componente luma de la matriz de predicción del macrobloque puede no estar completa, dado que para cada bloque luma  $4 \times 4$  el proceso de predicción `Intra_4x4` de muestras luma descrito en la subcláusula 8.3.1 y el proceso especificado en esta subcláusula se aplican reiteradamente.

Este proceso genera como resultado las matrices de muestras croma reconstruidas, antes de aplicar el proceso de filtrado de bloques de la componente que corresponda  $S'_L$ ,  $S'_{Cb}$ , o  $S'_{Cr}$ .

NOTA – Al decodificar un macrobloque en modo de predicción `Intra_4x4`, la componente luma de la matriz de predicción del macrobloque antes de aplicar el filtrado de bloque pueden no estar completas, dado que para cada bloque luma  $4 \times 4$  el proceso de predicción `Intra_4x4` de muestras luma descrito en la subcláusula 8.3.1 y el proceso especificado en esta subcláusula se aplican reiteradamente.

Esta subcláusula especifica el proceso de decodificación de coeficientes de transformada y el proceso de reconstrucción de imágenes antes de aplicar el proceso de filtrado de bloques.

Cuando el macrobloque considerado está codificado como `P_Skip` o `B_Skip`, todos los valores de `LumaLevel`, `ChromaDCLevel`, `ChromaACLevel` son iguales a 0 para el macrobloque considerado.

### 8.5.1 Especificación del proceso de decodificación de transformada para bloques residuales

Cuando en el modo de predicción de macrobloques considerado es distinto de `Intra_16x16`, la variable `LumaLevel` contiene los niveles de coeficientes de transformada luma. Para los bloques luma  $4 \times 4$  indexados mediante `luma4x4BlkIdx = 0..15`, se siguen los siguientes pasos.

1. Se llama al proceso de barrido inverso de coeficientes de transformada descrito en la subcláusula 8.5.4, pasándolo como argumento `LumaLevel[ luma4x4BlkIdx ]` y se obtiene como resultado la matriz bidimensional  $c$ .
2. Se llama al proceso de cambio de escala y de transformación para bloques  $4 \times 4$  residuales descrito en la subcláusula 8.5.8, pasándole  $c$  como argumento y como resultado se obtiene  $r$ .
3. Para calcular la posición de la muestra superior izquierda del bloque luma  $4 \times 4$  cuyo índice `luma4x4BlkIdx` está dentro del macrobloque se llama al proceso de barrido inverso de bloques luma  $4 \times 4$  descrito en la subcláusula 6.4.3, pasándole como argumento `luma4x4BlkIdx` y el resultado se asigna a  $(xO, yO)$ .
4. La matriz  $4 \times 4$   $u$ , con elementos  $u_{ij}$ , para  $i, j = 0..3$  se calcula del modo siguiente:

$$u_{ij} = \text{Clip1}(\text{pred}_L[xO + j, yO + i] + r_{ij}) \quad (8-243)$$



5. Se llama al proceso de reconstrucción de imágenes antes de aplicar el proceso de filtrado de bloques descrito en la subcláusula 8.5.9 pasándole como argumentos luma4x4BlkIdx y u, y se obtiene como resultado S'.

### 8.5.2 Especificación del proceso de decodificación de transformada para muestras luma de modo de predicción de macrobloque Intra\_16x16

Cuando el modo de predicción de macrobloque considerado es Intra\_16x16, las variables Intra16x16DCLevel e Intra16x16ACLevel contienen los niveles de los coeficientes de transformada luma. La decodificación de coeficientes de transformada consta de los siguientes pasos:

1. Se decodifican los coeficientes de transformada c.c. luma 4x4 de todos los bloques luma 4x4 del macrobloque.
  - a. Se llama al proceso de barrido inverso de los coeficientes de transformada descrito en la subcláusula 8.5.4, pasándole como argumento Intra16x16DCLevel, y se obtiene como resultado una matriz bidimensional c.
  - b. Se llama al proceso de cambio de escala y transformación de coeficientes de transformada c.c. luma para bloques del tipo Intra\_16x16, que se describen en la subcláusula 8.5.6, al que se le pasa como argumento c y el resultado es dcY.
2. Para cada bloque luma 4x4 indexado mediante luma4x4BlkIdx = 0..15, se siguen los siguientes pasos.
  - a. Se calcula la variable lumaList, que es una lista de 16 elementos. El primer elemento de lumaList es el valor correspondiente de la matriz dcY. En la figura 8-6 se muestra la asignación de los índices de la matriz dcY a luma4x4BlkIdx. Los dos números en los recuadros pequeños indican los índices i y j en dcY<sub>ij</sub>, y los números en los recuadros grandes indican luma4x4BlkIdx.

<sup>00</sup> 0	<sup>01</sup> 1	<sup>02</sup> 4	<sup>03</sup> 5
<sup>10</sup> 2	<sup>11</sup> 3	<sup>12</sup> 6	<sup>13</sup> 7
<sup>20</sup> 8	<sup>21</sup> 9	<sup>22</sup> 12	<sup>23</sup> 13
<sup>30</sup> 10	<sup>31</sup> 11	<sup>32</sup> 14	<sup>33</sup> 15

**Figura 8-6 – Asignación de los índices de dcY a luma4x4BlkIdx**

Los elementos en lumaList con índice k = 1..15, se definen del modo siguiente:

$$\text{lumaList}[k] = \text{Intra16x16ACLevel}[\text{luma4x4BlkIdx}][[k-1]] \quad (8-244)$$

- b. Se llama al proceso de barrido inverso de coeficiente de transformada descrito en la subcláusula 8.5.4, pasándole como argumento lumaList y el resultado que se obtiene es la matriz bidimensional c.
- c. Se llama al proceso de cambio de escala y transformación de bloques 4x4 residuales descrito en la subcláusula 8.5.8, pasándole como argumento c y el resultado es r.
- d. Para calcular la posición de la muestra superior izquierda del bloque luma 4x4 cuyo índice luma4x4BlkIdx está dentro del macrobloque se llama al proceso de barrido inverso de bloques luma 4x4, descrito en la subcláusula 6.4.3, al que se le pasa como argumento luma4x4BlkIdx y el resultado se asigna a ( xO, yO ).
- e. Se calcula la matriz 4x4 u, con elementos u<sub>ij</sub>, para i, j = 0..3.

$$u_{ij} = \text{Clip1}(\text{pred}_i[xO + j, yO + i] + r_{ij}) \quad (8-245)$$

- f. Se llama al proceso de reconstrucción de imágenes antes de aplicar el proceso de filtrado de bloques descrito en la subcláusula 8.5.9, pasándole como argumentos luma4x4BlkIdx y u, y el resultado es S'.

### 8.5.3 Especificación del proceso de decodificación de transformada para muestras croma

Para cada componente croma, las variables ChromaDCLevel[ iCbCr ] y ChromaACLevel[ iCbCr ], siendo iCbCr igual a 0 para Cb e iCbCr igual a 1 para Cr, contienen los niveles de las dos componentes de los coeficientes de transformada croma. El proceso de codificación de transformada se aplica por separado a cada componente croma, para lo cual se siguen los siguientes pasos:

1. Se decodifican los coeficientes de transformada c.c. croma 2x2 de los bloques croma 4x4 de la componente indexada mediante iCbCr del macrobloque.
  - a. Se calcula la matriz 2x2 c utilizando el proceso de barrido por líneas inverso aplicado a ChromaDCLevel del modo siguiente:

$$c = \begin{bmatrix} \text{ChromaDCLevel}[iCbCr][0] & \text{ChromaDCLevel}[iCbCr][1] \\ \text{ChromaDCLevel}[iCbCr][2] & \text{ChromaDCLevel}[iCbCr][3] \end{bmatrix} \quad (8-246)$$

- b. Se llama al proceso de cambio de escala y transformación de coeficientes de transformada c.c. croma, descrito en la subcláusula 8.5.7, pasándole como argumento c y el resultado es dcC.
2. A cada bloque croma 4x4 indexado mediante chroma4x4BlkIdx = 0..3 de la componente indexada mediante iCbCr, se aplican los siguientes pasos.
  - a. Se calcula la variable chromaList, que es una lista de 16 elementos. El primer elemento de chromaList es el valor correspondiente de la matriz dcC. La figura 8-7 ilustra la asignación de índices de la matriz dcC a el chroma4x4BlkIdx. Los dos números en los recuadros pequeños indican las componentes i y j de dcC<sub>ij</sub>, y los números en los cuadrados grandes indican el chroma4x4BlkIdx.

00	01
0	1
10	11
2	3

**Figura 8-7 – Asignación de los índices dcC a chroma4x4BlkIdx**

Los elementos de chromaList con índice k = 1..15 se definen como

$$\text{chromaList}[k] = \text{ChromaACLevel}[\text{chroma4x4BlkIdx}][k - 1] \quad (8-247)$$

- b. Se llama al proceso de barrido inverso de coeficientes de transformada, descrito en la subcláusula 8.5.4, pasándole como argumento chromaList y el resultado es la matriz bidimensional c.
  - c. Se llama al proceso de cambio de escala y transformación de bloques 4x4 residuales, descrito en la subcláusula 8.5.8, pasándole como argumento c y el resultado es r.
  - d. Se calcula la posición de la muestra superior izquierda del bloque croma 4x4 cuyo índice es chroma4x4BlkIdx dentro del macrobloque, del modo siguiente:

$$xO = \text{InverseRasterScan}(\text{chroma4x4BlkIdx}, 4, 4, 8, 0) \quad (8-248)$$

$$yO = \text{InverseRasterScan}(\text{chroma4x4BlkIdx}, 4, 4, 8, 1) \quad (8-249)$$

- e. Se calcula la matriz 4x4 u de elementos u<sub>ij</sub>, para i, j = 0..3

$$u_{ij} = \text{Clip1}(\text{pred}_c[ xO + j, yO + i ] + r_{ij}) \quad (8-250)$$

- f. Se llama al proceso de reconstrucción de imágenes antes de aplicar el proceso de filtrado de bloques, descrito en la subcláusula 8.5.9, pasándole como argumentos `chroma4x4BlkIdx` y `u`, y el resultado es `S'`.

### 8.5.4 Proceso de barrido inverso de coeficientes de transformada

Este proceso acepta como argumentos una lista de 16 valores.

Este proceso genera como resultado una variable `c` que contiene una matriz bidimensional de 4x4 valores con los niveles asignados a posiciones en el bloque de transformada.

El proceso de decodificación mapea la secuencia de niveles de coeficientes de transformada en posiciones de niveles de coeficientes de transformada. Para realizar este mapeado se utilizan los dos patrones de barrido inverso que ilustra la figura 8-8.

El barrido inverso en zig-zag se utiliza para macrobloques cuadro y el barrido inverso por campo se utiliza para macrobloques campo.

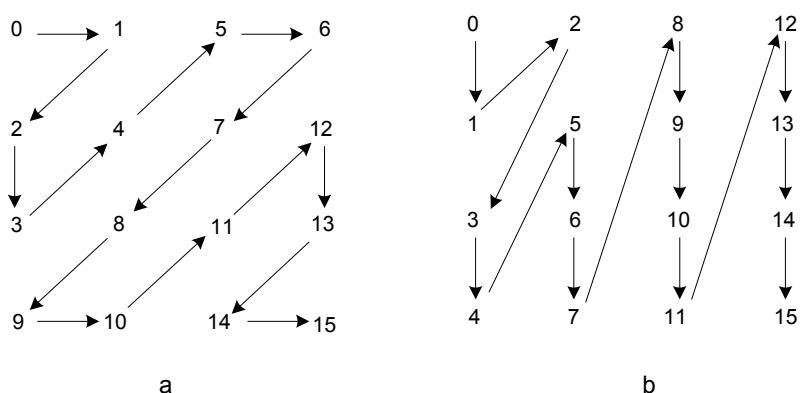


Figura 8-8 – a) Barrido en Zig-zag. b) Barrido por campo

En el cuadro 8-12 se describe el mapeado del índice `idx` de la lista de 16 elementos en índices `i` y `j` de la matriz bidimensional `c`.

Cuadro 8-12 – Especificación del mapeado de `idx` en `cij` para el barrido en zig-zag y por campo

idx	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
zig-zag	<code>c<sub>00</sub></code>	<code>c<sub>01</sub></code>	<code>c<sub>10</sub></code>	<code>c<sub>20</sub></code>	<code>c<sub>11</sub></code>	<code>c<sub>02</sub></code>	<code>c<sub>03</sub></code>	<code>c<sub>12</sub></code>	<code>c<sub>21</sub></code>	<code>c<sub>30</sub></code>	<code>c<sub>31</sub></code>	<code>c<sub>22</sub></code>	<code>c<sub>13</sub></code>	<code>c<sub>23</sub></code>	<code>c<sub>32</sub></code>	<code>c<sub>33</sub></code>
campo	<code>c<sub>00</sub></code>	<code>c<sub>10</sub></code>	<code>c<sub>01</sub></code>	<code>c<sub>20</sub></code>	<code>c<sub>30</sub></code>	<code>c<sub>11</sub></code>	<code>c<sub>21</sub></code>	<code>c<sub>31</sub></code>	<code>c<sub>02</sub></code>	<code>c<sub>12</sub></code>	<code>c<sub>22</sub></code>	<code>c<sub>32</sub></code>	<code>c<sub>03</sub></code>	<code>c<sub>13</sub></code>	<code>c<sub>23</sub></code>	<code>c<sub>33</sub></code>

### 8.5.5 Proceso de cálculo de parámetros de cuantificación croma y función de cambio de escala

Este proceso genera como resultados:

- `QPC`: el parámetro de cuantificación croma.
- `QSC`: el parámetro de cuantificación croma adicional necesario para decodificar sectores SP y SI (si los hubiere)

NOTA – Los valores de los parámetros de cuantificación `QP`, `QPY`, `QPC`, `QSY` y `QSC` están siempre entre 0 y 51, inclusive.

El valor de `QPC` para croma se determina a partir del valor actual de `QPY` y el valor de `chroma_qp_index_offset`.

NOTA – Las ecuaciones de cambio de escala se especifican de modo que el factor de escala del nivel de coeficiente de transformada equivalente sea el doble para cada incremento en 6 de  $QP_Y$ . Así pues, el factor utilizado para el cambio de escala aumenta aproximadamente un 12 % por cada aumento en 1 del valor de  $QP_Y$ .

El valor de  $QP_C$  en función del índice  $qP_I$  se determina mediante el cuadro 8-13. El valor de  $qP_I$  se calcula del modo siguiente.

$$qP_I = \text{Clip3}(0, 51, QP_Y + \text{chroma\_qp\_index\_offset}) \quad (8-251)$$

**Cuadro 8-13 – Especificación de  $QP_C$  como una función de  $qP_I$**

$qP_I$	<30	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51
$QP_C$	= $qP_I$	29	30	31	32	32	33	34	34	35	35	36	36	37	37	37	38	38	38	39	39	39	39

Cuando el sector considerado es un sector SP o SI,  $QS_C$  se calcula utilizando el proceso anterior y sustituyendo  $QP_Y$  por  $QS_Y$  y  $QP_C$  por  $QS_C$ .

La función  $\text{LevelScale}(m, i, j)$  se define del modo siguiente:

$$\text{LevelScale}(m,i,j) = \begin{cases} v_{m0} & \text{para } (i,j) \in \{(0,0), (0,2), (2,0), (2,2)\}, \\ v_{m1} & \text{para } (i,j) \in \{(1,1), (1,3), (3,1), (3,3)\}, \\ v_{m2} & \text{en otro caso} \end{cases} \quad (8-252)$$

donde el primer y segundo subíndices de  $v$  son, respectivamente, los índices de fila y de columna de la siguiente matriz:

$$v = \begin{bmatrix} 10 & 16 & 13 \\ 11 & 18 & 14 \\ 13 & 20 & 16 \\ 14 & 23 & 18 \\ 16 & 25 & 20 \\ 18 & 29 & 23 \end{bmatrix}. \quad (8-253)$$

### 8.5.6 Proceso de cambio de escala y de transformación de coeficientes de transformada c.c. luma para macrobloques del tipo Intra\_16x16

Este proceso acepta como argumento los valores de los niveles de coeficientes de transformada para coeficientes de transformada c.c. luma de macrobloques Intra\_16x16, que son una matriz 4x4  $c$  con elementos  $c_{ij}$ , donde  $i$  y  $j$  forman un índice de frecuencia bidimensional.

Este proceso genera como resultado 16 valores c.c. a escala para bloques 4x4 luma de macrobloques Intra\_16x16, en la forma de una matriz 4x4  $dcY$  con elementos  $dcY_{ij}$ .

La transformada inversa de los coeficientes de transformada c.c. luma 4x4 es la siguiente:

$$f = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} c_{00} & c_{01} & c_{02} & c_{03} \\ c_{10} & c_{11} & c_{12} & c_{13} \\ c_{20} & c_{21} & c_{22} & c_{23} \\ c_{30} & c_{31} & c_{32} & c_{33} \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix}. \quad (8-254)$$

Los trenes de bits conformes con esta Recomendación | Norma Internacional no contendrán datos que den lugar a elementos  $f_{ij}$  de  $f$  que sobrepasen los valores enteros  $-2^{15}$  y  $2^{15}-1$ , inclusive.

Después de aplicar la transformada inversa, el cambio de escala se realiza del modo siguiente:

- Si  $QP_Y$  es mayor o igual a 12, el cambio de escala se calcula mediante la expresión:

$$dcY_{ij} = (f_{ij} * \text{LevelScale}(QP_Y \% 6, 0, 0)) \ll (QP_Y / 6 - 2), \quad \text{con } i, j = 0..3 \quad (8-255)$$

- De lo contrario ( $QP_Y$  es menor que 12) el cambio de escala se realiza mediante la expresión:

$$dcY_{ij} = (f_{ij} * \text{LevelScale}(QP_Y \% 6, 0, 0) + 2^{1-QP_Y/6}) \gg (2 - QP_Y / 6), \quad \text{siendo } i, j = 0..3 \quad (8-256)$$

Los trenes de bits conformes con esta Recomendación | Norma Internacional no contendrán datos que den lugar a que algún elemento  $dcY_{ij}$  de  $dcY$  supere los valores enteros  $-2^{15}$  y  $2^{15}-1$ , inclusive.

NOTA – Cuando `entropy_coding_mode_flag` es igual a 0 y  $QP_Y$  es menor que 10, la gama de valores que pueden representarse para los elementos  $c_{ij}$  de  $c$  es insuficiente para representar la gama entera de valores de los elementos  $dcY_{ij}$  de  $dcY$  que podrían ser necesarios para generar una aproximación cercana al contenido de cualquier imagen original posible mediante la utilización del tipo de macrobloque `Intra_16x16`.

NOTA – Debido a que el límite de la gama que se fija para los elementos  $dcY_{ij}$  de  $dcY$  se fija después del corrimiento a la derecha en la ecuación 8-256, se debe tolerar una mayor gama de valores en el decodificador antes del corrimiento a la derecha.

### 8.5.7 Proceso de cambio de escala y transformación de coeficientes de transformada c.c. croma

Este proceso acepta como argumentos valores de nivel de coeficientes de transformadas para coeficientes de transformada c.c. croma de una componente croma del macrobloque, en la forma de una matriz  $2 \times 2$   $c$  de elementos  $c_{ij}$ , donde  $i$  y  $j$  forman un índice de frecuencias bidimensional.

Este proceso genera como resultado los cuatro valores c.c. a escala de una matriz  $2 \times 2$   $dcC$  de elementos  $dcC_{ij}$ .

La transformada inversa de coeficientes de transformada c.c. croma  $2 \times 2$  se define del modo siguiente:

$$f = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} c_{00} & c_{01} \\ c_{10} & c_{11} \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \quad (8-257)$$

Los trenes de bits conformes con esta Recomendación | Norma Internacional no contendrán datos que den lugar a elementos  $f_{ij}$  de  $f$  cuyo valor entero no esté comprendido entre  $-2^{15}$  y  $2^{15}-1$ , inclusive.

Después de aplicar la transformada inversa, el cambio de escala se realiza del modo siguiente:

- Si  $QP_C$  es mayor o igual a 6, el cambio de escala se calcula mediante la siguiente expresión:

$$dcC_{ij} = (f_{ij} * \text{LevelScale}(QP_C \% 6, 0, 0)) \ll (QP_C / 6 - 1), \quad \text{con } i, j = 0, 1 \quad (8-258)$$

- De lo contrario ( $QP_C$  es menor que 6), el cambio de escala se calcula del modo siguiente:

$$dcC_{ij} = (f_{ij} * \text{LevelScale}(QP_C \% 6, 0, 0)) \gg 1, \quad \text{con } i, j = 0, 1 \quad (8-259)$$

Los trenes bits conformes con esta Recomendación | Norma Internacional no contendrán datos que den lugar a elementos  $dcC_{ij}$  de  $dcC$  cuyo valor entero no esté comprendido entre  $-2^{15}$  y  $2^{15}-1$ , inclusive.

NOTA – Cuando `entropy_coding_mode_flag` es igual a 0 y  $QP_Y$  es menor que 4, la gama de valores que pueden representarse para los elementos  $c_{ij}$  de  $c$  es insuficiente para representar la gama entera de valores de los elementos  $dcC_{ij}$  de  $dcC$  que podrían ser necesarios para generar una aproximación cercana al contenido de cualquier imagen original.

NOTA – Debido a que el límite de la gama que se fija para los elementos  $dcC_{ij}$  de  $dcC$  se fija después del corrimiento a la derecha en la ecuación 8-259, se debe tolerar una mayor gama de valores en el decodificador antes del corrimiento a la derecha.

### 8.5.8 Proceso de cambio de escala y transformación de bloques 4x4 residuales

Este proceso acepta como argumentos la matriz 4x4  $c$  de elementos  $c_{ij}$ , que es una matriz correspondiente a un bloque residual de la componente luma o a un bloque residual de una componente croma.

Este proceso genera como resultado valores de muestras residuales, en la forma de una matriz 4x4  $r$  de elementos  $r_{ij}$ .

La variable  $sMbFlag$  se calcula del modo siguiente.

- Si  $mb\_type$  es igual a SI o el modo de predicción de macrobloque es inter en un sector SP,  $sMbFlag$  se pone a 1,
- De lo contrario ( $mb\_type$  no es igual a SI y el modo de predicción de macrobloque no es Inter en un sector SP),  $sMbFlag$  se pone a 0.

La variable  $qP$  se calcula del modo siguiente.

- Si la matriz  $c$  pasada como argumento corresponde a un bloque residual luma y  $sMbFlag$  es igual a 0

$$qP = QP_Y \quad (8-260)$$

- De lo contrario, si la matriz  $c$  pasada como argumento corresponde a un bloque residual luma y  $sMbFlag$  es igual a 1

$$qP = QS_Y \quad (8-261)$$

- De lo contrario, si la matriz  $c$  pasada como argumento corresponde a un bloque residual croma y  $sMbFlag$  es igual a 0

$$qP = QP_C \quad (8-262)$$

- De lo contrario (si la matriz  $c$  pasada como argumento corresponde a un bloque residual croma y  $sMbFlag$  es igual a 1),

$$qP = QS_C \quad (8-263)$$

El cambio de escala de los niveles de coeficiente de transformada de bloques 4x4 de elementos  $C_{ij}$  se realiza del modo siguiente.

- Si se cumplen todas las condiciones siguientes:
  - $i$  es igual a 0
  - $j$  es igual a 0
  - $c$  corresponde a un bloque residual luma codificado en modo de predicción Intra\_16x16 o  $c$  corresponde a un bloque residual croma

la variable  $d_{00}$  es:

$$d_{00} = c_{00} \quad (8-264)$$

- De lo contrario,

$$d_{ij} = (c_{ij} * LevelScale(qP \% 6, i, j)) \ll (qP / 6), \quad \text{con } i, j = 0..3 \text{ salvo lo indicado antes} \quad (8-265)$$

El tren de bits no contendrá datos que den lugar a que algún elemento  $d_{ij}$  de  $d$  con  $i, j = 0..3$  tenga un valor entero que no esté comprendido entre  $-2^{15}$  y  $2^{15}-1$ , inclusive.

La aplicación de la transformada convertirá el bloque de coeficientes de transformada a escala en un bloque de muestras resultantes matemáticamente equivalente al siguiente.

En primer lugar, se aplica a cada fila (horizontal) de coeficientes de transformada a escala una transformada inversa unidimensional, como se indica a continuación.

El conjunto de valores intermedios se calcula del modo siguiente.

$$e_{i0} = d_{i0} + d_{i2}, \text{ con } i = 0..3 \quad (8-266)$$

$$e_{i1} = d_{i0} - d_{i2}, \text{ con } i = 0..3 \quad (8-267)$$

$$e_{i2} = (d_{i1} \gg 1) - d_{i3}, \text{ con } i = 0..3 \quad (8-268)$$

$$e_{i3} = d_{i1} + (d_{i3} \gg 1), \text{ con } i = 0..3 \quad (8-269)$$

El tren de bits no contendrá datos que den lugar a que algún elemento  $e_{ij}$  de  $e$  para  $i, j = 0..3$  tenga un valor entero que no esté comprendido entre  $-2^{15}$  y  $2^{15}-1$ , inclusive.

Así pues, el resultado se calcula a partir de estos valores intermedios.

$$f_{i0} = e_{i0} + e_{i3}, \text{ con } i = 0..3 \quad (8-270)$$

$$f_{i1} = e_{i1} + e_{i2}, \text{ con } i = 0..3 \quad (8-271)$$

$$f_{i2} = e_{i1} - e_{i2}, \text{ con } i = 0..3 \quad (8-272)$$

$$f_{i3} = e_{i0} - e_{i3}, \text{ con } i = 0..3 \quad (8-273)$$

El tren de bits no contendrá datos que den lugar a que algún elemento  $f_{ij}$  de  $f$  para  $i, j = 0..3$  tenga un valor entero que no esté comprendido entre  $-2^{15}$  y  $2^{15}-1$ , inclusive.

A continuación, se aplica a cada columna (vertical) de la matriz resultante una transformada inversa unidimensional que se describe a continuación.

El conjunto de valores intermedios se calcula del modo siguiente.

$$g_{0j} = f_{0j} + f_{2j}, \text{ con } j = 0..3 \quad (8-274)$$

$$g_{1j} = f_{0j} - f_{2j}, \text{ con } j = 0..3 \quad (8-275)$$

$$g_{2j} = (f_{1j} \gg 1) - f_{3j}, \text{ con } j = 0..3 \quad (8-276)$$

$$g_{3j} = f_{1j} + (f_{3j} \gg 1), \text{ con } j = 0..3 \quad (8-277)$$

El tren de bits no contendrá datos que den lugar a que algún elemento  $g_{ij}$  de  $g$  para  $i, j = 0..3$  tenga un valor entero que no esté comprendido entre  $-2^{15}$  y  $2^{15}-1$ , inclusive.

Seguidamente, se calcula el resultado a partir de estos valores intermedios.

$$h_{0j} = g_{0j} + g_{3j}, \text{ con } j = 0..3 \quad (8-278)$$

$$h_{1j} = g_{1j} + g_{2j}, \text{ con } j = 0..3 \quad (8-279)$$

$$h_{2j} = g_{1j} - g_{2j}, \text{ con } j = 0..3 \quad (8-280)$$

$$h_{3j} = g_{0j} - g_{3j}, \text{ con } j = 0..3 \quad (8-281)$$

El tren de bits no contendrá datos que den lugar a que algún elemento  $h_{ij}$  de  $h$  para  $i, j = 0..3$  tenga un valor entero que no esté comprendido entre  $-2^{15}$  y  $2^{15}-33$ , inclusive.

Por último, después de aplicar las dos transformadas inversas unidimensionales horizontal y vertical para obtener la matriz de muestras transformadas, los valores de la muestra residuales reconstruidas se calculan mediante la expresión:

$$r_{ij} = (h_{ij} + 2^5) \gg 6 \text{ con } i, j = 0..3 \quad (8-282)$$

### 8.5.9 Proceso de reconstrucción de imágenes antes de aplicar el proceso de filtrado de bloques

Este proceso acepta como argumentos

- luma4x4BlkIdx o chroma4x4BlkIdx;
- una matriz  $u$   $4 \times 4$  de muestras residuales reconstruidas de elementos  $u_{ij}$ , que es un bloque residual luma o croma.

Este proceso genera como resultado bloques de muestras reconstruidas antes de aplicar el proceso de filtrado de bloques.

Para calcular la posición de la muestra luma superior izquierda del macrobloque considerado se llama al proceso de barrido inverso de macrobloques descrito en la subcláusula 6.4.1, al que se le pasa como argumento CurrMbAddr y el resultado se asigna a ( xP, yP ).

Cuando u es un bloque luma, a cada muestra  $u_{ij}$  del bloque luma 4x4 se aplica lo siguiente.

- Para calcular la posición de la muestra superior izquierda de un bloque luma 4x4 con índice luma4x4BlkIdx dentro del macrobloque se llama al proceso de barrido inverso de bloques luma 4x4 descrito en la subcláusula 6.4.3, al que se le pasa como argumento luma4x4BlkIdx y el resultado se asigna a ( xO, yO ).

- En función de la variable MbaffFrameFlag, se aplica lo siguiente.

- Si MbaffFrameFlag es igual a 1 y el macrobloque considerado es un macrobloque campo

$$S'_L[ xP + xO + j, yP + 2 * ( yO + i ) ] = u_{ij} \quad \text{con } i, j = 0..3 \quad (8-283)$$

- De lo contrario (MbaffFrameFlag es igual a 0 y el macrobloque considerado es un macrobloque cuadro),

$$S'_L[ xP + xO + j, yP + yO + i ] = u_{ij} \quad \text{con } i, j = 0..3 \quad (8-284)$$

Cuando u es un bloque croma, a cada muestra  $u_{ij}$  del bloque croma 4x4 se aplica lo siguiente.

- El subíndice C en las variables  $S'_C$  y  $pred_C$  se sustituye por Cb para la componente croma Cb y por Cr para la componente croma Cr.

- Se calcula la posición de la muestra superior izquierda del bloque croma 4x4 con índice chroma4x4BlkIdx dentro del macrobloque.

$$xO = \text{InverseRasterScan}( \text{chroma4x4BlkIdx}, 4, 4, 8, 0 ) \quad (8-285)$$

$$yO = \text{InverseRasterScan}( \text{chroma4x4BlkIdx}, 4, 4, 8, 1 ) \quad (8-286)$$

- En función de la variable MbaffFrameFlag, se aplica lo siguiente.

- Si MbaffFrameFlag es igual a 1 y el macrobloque considerado es un macrobloque campo.

$$S'_C[ ( xP \gg 1 ) + xO + j, ( ( yP + 1 ) \gg 1 ) + 2 * ( yO + i ) ] = u_{ij} \quad \text{con } i, j = 0..3 \quad (8-287)$$

- De lo contrario (MbaffFrameFlag es igual a 0 o el macrobloque considerado es un macrobloque cuadro),

$$S'_C[ ( xP \gg 1 ) + xO + j, ( ( yP + 1 ) \gg 1 ) + yO + i ] = u_{ij} \quad \text{con } i, j = 0..3 \quad (8-288)$$

## 8.6 Proceso de decodificación de macrobloques P en sectores SP o macrobloques SI

Se llama a este proceso para decodificar macrobloques de tipo P en sectores SP o macrobloques de tipo SI en sectores SI.

Este proceso acepta como argumentos niveles de coeficientes de transformada residuales de predicción y las muestras predichas correspondientes al macrobloque considerado.

Este proceso genera como resultado las muestras decodificadas del macrobloque considerado antes de aplicar el proceso de filtrado de bloques.

Esta subcláusula especifica el proceso de decodificación de coeficientes de transformada y el proceso de reconstrucción de imágenes de macrobloques de tipo P en sectores SP y macrobloques de tipo SI en sectores SI.

NOTA – Los sectores SP utilizan la codificación predictiva inter para explotar la redundancia temporal de la secuencia, de manera similar a la codificación de sectores P. Sin embargo, a diferencia de ésta, la codificación de sectores SP permite la reconstrucción idéntica de un sector aun cuando se utilicen imágenes de referencia diferentes. En los sectores SI se utiliza la predicción espacial de manera similar a como se utiliza en sectores I. La codificación de sectores SI permite a reconstrucción idéntica a un sector SP correspondiente. Las propiedades de los sectores SP y SI



ayudan a facilitar funcionalidades para la conmutación, concatenación, acceso aleatorio, avance rápido, retroceso rápido, y robustez/recuperación de errores en trenes de bits.

Un sector SP consta de macrobloques codificados como macrobloques de tipo I o de tipo P.

Un sector SI consta de macrobloques codificados como macrobloques de tipo I o de tipo SI.

Se llamará a los procesos de decodificación de coeficientes de transformada y de reconstrucción de imágenes antes de aplicar el proceso de filtrado de bloques para macrobloques del tipo I en sectores SI, como se describe en la subcláusula 8.5. Los macrobloques del tipo SI se decodificarán según se describe a continuación.

Cuando el macrobloque considerado está codificado como P\_Skip, todos los valores de LumaLevel, ChromaDCLevel, ChromaACLevel de ese macrobloque se ponen a 0.

### 8.6.1 Proceso de decodificación de sectores SP de imágenes no intercambiadas

Se llama a este proceso al decodificar macrobloques del tipo P en sectores SP para los cuales `sp_for_switch_flag` es igual a 0.

Este proceso acepta como argumentos muestras de predicción Inter del macrobloque considerado según se describe en la subcláusula 8.4 y niveles de coeficientes de transformada residuales de predicción.

Este proceso genera como resultado muestras decodificadas del macrobloque considerado antes de aplicar el proceso de filtrado de bloques.

Esta subcláusula se aplica a todos los macrobloques en sectores SP para los cuales `sp_for_switch_flag` es igual a 0, excepto a aquellos cuyo modo de predicción de macrobloque sea `Intra_4x4` o `Intra_16x16`. No se aplica a sectores SI.

#### 8.6.1.1 Proceso de decodificación de coeficientes de transformada luma

Este proceso acepta como argumentos muestras luma de predicción inter del macrobloque considerado  $pred_L$ , descritas en la subcláusula 8.4, los niveles de coeficientes de transformada residual de predicción, LumaLevel, y el índice del bloque luma 4x4 `luma4x4BlkIdx`.

Este proceso genera como resultado las muestras luma decodificadas del macrobloque considerado antes de aplicar el proceso de filtrado de bloques.

Para calcular la posición de la muestra superior izquierda del bloque luma 4x4 con índice `luma4x4BlkIdx` dentro del macrobloque considerado se llama al proceso de barrido inverso de bloques luma 4x4, descrito en la subcláusula 6.4.3, al que se le pasa como argumento `luma4x4BlkIdx` y el resultado se asigna a  $(x, y)$ .

Sea la variable  $p$  una matriz 4x4 de muestras de predicción de elementos  $p_{ij}$  que se calculan del modo siguiente.

$$p_{ij} = pred_L[x + j, y + i] \quad \text{con } i, j = 0..3 \quad (8-289)$$

A partir de la variable  $p$  se obtienen los coeficientes de transformada  $c^p$  mediante la siguiente operación:

$$c^p = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} \begin{bmatrix} p_{00} & p_{01} & p_{02} & p_{03} \\ p_{10} & p_{11} & p_{12} & p_{13} \\ p_{20} & p_{21} & p_{22} & p_{23} \\ p_{30} & p_{31} & p_{32} & p_{33} \end{bmatrix} \begin{bmatrix} 1 & 2 & 1 & 1 \\ 1 & 1 & -1 & -2 \\ 1 & -1 & -1 & 2 \\ 1 & -2 & 1 & -1 \end{bmatrix} \quad (8-290)$$

Se llama al proceso de barrido inverso de coeficiente de transformada descrito en la subcláusula 8.5.4, pasándole como argumento `LumaLevel[luma4x4BlkIdx]` y se obtiene como resultado la matriz bidimensional  $c^t$ , de elementos  $c_{ij}^t$ .

Se aplica un cambio de escala a los coeficientes de transformada residuales de predicción  $c^r$  mediante el parámetro de cuantificación  $QP_Y$ , y se suman a los coeficientes de transformada del bloque de predicción  $c^p$ , con  $i, j = 0..3$ , mediante la expresión siguiente.

$$c_{ij}^s = c_{ij}^p + ( ( ( c_{ij}^r * \text{LevelScale}(QP_Y \% 6, i, j) * A_{ij} ) \ll (QP_Y / 6) ) \gg 6 ) \quad (8-291)$$

donde  $\text{LevelScale}(m, i, j)$  viene dado por la ecuación 8-252 y  $A_{ij}$  es:

$$A_{ij} = \begin{cases} 16 & \text{para } (i, j) \in \{(0,0), (0,2), (2,0), (2,2)\}, \\ 25 & \text{para } (i, j) \in \{(1,1), (1,3), (3,1), (3,3)\}, \\ 20 & \text{en otro caso} \end{cases} \quad (8-292)$$

La función  $\text{LevelScale2}(m, i, j)$  utilizada en las fórmulas anteriores se define del modo siguiente:

$$\text{LevelScale2}(m, i, j) = \begin{cases} w_{m0} & \text{para } (i, j) \in \{(0,0), (0,2), (2,0), (2,2)\}, \\ w_{m1} & \text{para } (i, j) \in \{(1,1), (1,3), (3,1), (3,3)\}, \\ w_{m2} & \text{en otro caso;} \end{cases} \quad (8-293)$$

donde el primero y segundo subíndices de  $w$  son, respectivamente, los índices de fila y columna de la siguiente matriz:

$$w = \begin{bmatrix} 13107 & 5243 & 8066 \\ 11916 & 4660 & 7490 \\ 10082 & 4194 & 6554 \\ 9362 & 3647 & 5825 \\ 8192 & 3355 & 5243 \\ 7282 & 2893 & 4559 \end{bmatrix} \quad (8-294)$$

La suma resultante  $c^s$  se cuantifica mediante el parámetro de cuantificación  $QS_Y$ , siendo  $i, j = 0..3$ .

$$c_{ij} = ( \text{Sign}(c_{ij}^s) * ( \text{Abs}(c_{ij}^s) * \text{LevelScale2}(QS_Y \% 6, i, j) + (1 \ll (14 + QS_Y / 6)) ) ) \gg (15 + QS_Y / 6) \quad (8-295)$$

Se llama al proceso de cambio de escala y de transformación para bloques 4x4 residuales descrito en la subcláusula 8.5.8, pasándole como argumento  $c$  y se obtiene como resultado  $r$ .

La matriz 4x4  $u$ , de elementos  $u_{ij}$ , se calcula del modo siguiente.

$$u_{ij} = \text{Clip1}(r_{ij}) \text{ con } i, j = 0..3 \quad (8-296)$$

Se llama al proceso de reconstrucción de imágenes antes de aplicar el proceso de filtrado de bloques descrito en la subcláusula 8.5.9, pasándole como argumento  $\text{luma4x4BlkIdx}$  y se obtiene como resultado  $S'$ .

### 8.6.1.2 Proceso de decodificación de coeficientes de transformada croma

Este proceso acepta como argumentos muestras croma de predicción inter del macrobloque considerado, descrito en la subcláusula 8.4, y los niveles de coeficientes de transformada residuales de predicción,  $\text{ChromaDCLevel}$  y  $\text{ChromaACLevel}$ .

Este proceso genera como resultado muestras croma decodificadas del macrobloque considerado antes de aplicar el proceso de filtrado de bloques.

Se llama dos veces a este proceso: una vez para la componente  $C_b$  y otra para la componente  $C_r$ . Para referirse a la componente  $C_b$  se sustituye  $C$  por  $C_b$  y para referirse a la componente  $C_r$  se sustituye  $C$  por  $C_r$ . Sea  $iCbCr$  la componente croma considerada.

A cada bloque 4x4 de la componente croma considerada con índice  $\text{chroma4x4BlkIdx}$ , siendo  $\text{chroma4x4BlkIdx}$  igual a 0..3, se aplica lo siguiente.

- Se calcula la posición de la muestra superior izquierda del bloque croma 4x4 con índice  $\text{chroma4x4BlkIdx}$  dentro del macrobloque.

$$x = \text{InverseRasterScan}(\text{chroma4x4BlkIdx}, 4, 4, 8, 0) \quad (8-297)$$

$$y = \text{InverseRasterScan}(\text{chroma4x4BlkIdx}, 4, 4, 8, 1) \quad (8-298)$$

– Se calcula la matriz 4x4 p de muestras de predicción, con elementos  $p_{ij}$ .

$$p_{ij} = \text{predc}[x + j, y + i] \quad \text{con } i, j = 0..3 \quad (8-299)$$

– Se aplica la ecuación 8-290 a la matriz 4x4 p para obtener los coeficientes de transformada  $c^p(\text{chroma4x4BlkIdx})$ .

– Se calcula la variable chromaList, que es una lista de 16 elementos. chromaList[0] se pone a 0. chromaList[k] con  $k = 1..15$  se calcula del modo siguiente.

$$\text{chromaList}[k] = \text{ChromaACLevel}[\text{iCbCr}][\text{chroma4x4BlkIdx}][k - 1] \quad (8-300)$$

– Se llama al proceso de barrido inverso de coeficiente de transformada descrito en la subcláusula 8.5.4, pasándole como argumento chromaList y se obtiene como resultado la matriz 4x4  $c^r$ .

– Se aplica un cambio de escala a los coeficientes de transformada residual de predicción  $c^r$  mediante el parámetro de cuantificación  $QP_C$ , y los coeficientes resultantes se suman a los coeficientes de transformada del bloque de predicción  $c^p$ , con  $i, j = 0..3$ , salvo para la combinación  $i = 0, j = 0$ .

$$c_{ij}^s = c_{ij}^p(\text{chroma4x4BlkIdx}) + (((c_{ij}^r * \text{LevelScale}(QP_C \% 6, i, j) * A_{ij}) \ll (QP_C / 6)) \gg 6) \quad (8-301)$$

– La suma resultante  $c^s$  se cuantifica mediante el parámetro de cuantificación  $QS_C$ , siendo  $i, j = 0..3$  excepto para la combinación  $i = 0, j = 0$ . El cálculo de  $c_{00}(\text{chroma4x4BlkIdx})$  se describe más adelante en esta subcláusula.

$$c_{ij}(\text{chroma4x4BlkIdx}) = \frac{(\text{Sign}(c_{ij}^s) * (\text{Abs}(c_{ij}^s) * \text{LevelScale2}(QS_C \% 6, i, j) + (1 \ll (14 + QS_C / 6)))) \gg (15 + QS_C / 6)}{\quad} \quad (8-302)$$

– Se llama al proceso de cambio de escala y transformación de bloques 4x4 residuales descrito en la subcláusula 8.5.8, pasándole como argumento  $c(\text{chroma4x4BlkIdx})$  y se obtiene como resultado r.

– Se calcula la matriz 4x4 u, de elementos  $u_{ij}$ .

$$u_{ij} = \text{Clip1}(r_{ij}) \quad \text{con } i, j = 0..3 \quad (8-303)$$

– Se llama al proceso de reconstrucción de imágenes antes de aplicar el proceso de filtrado de bloques descrito en la subcláusula 8.5.9, pasándole como argumentos chroma4x4BlkIdx y u, se obtiene como resultado  $S'$ .

A continuación se describe el cálculo del nivel de coeficientes de transformada c.c.  $c_{00}(\text{chroma4x4BlkIdx})$ . Los coeficientes de transformada c.c. de los 4 bloques 4x4 croma de predicción de la componente considerada del macrobloque se agrupan en una matriz 2x2 de elementos  $c_{00}^p(\text{chroma4x4BlkIdx})$  y se aplica la siguiente transformada 2x2 a los coeficientes de transformada c.c..

$$dc^p = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} c_{00}^p(0) & c_{00}^p(1) \\ c_{00}^p(2) & c_{00}^p(3) \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \quad (8-304)$$

Se aplica un cambio de escala a los niveles de coeficientes de transformada residuales de predicción c.c. croma, ChromaDCLevel[iCbCr][k] con  $k = 0..3$ , mediante el parámetro de cuantificación  $QP$ , y se suman a los coeficientes resultantes los coeficientes de transformada c.c. de predicción.

$$dc_{ij}^s = dc_{ij}^p + (((\text{ChromaDCLevel}[\text{iCbCr}][j * 2 + i] * \text{LevelScale}(QP_C \% 6, 0, 0) * A_{00}) \ll (QP_C / 6)) \gg 5) \quad \text{con } i, j = 0, 1 \quad (8-305)$$

La matriz 2x2  $dc^s$  se cuantifica mediante el parámetro de cuantificación,  $QS_C$ .

$$dc_{ij}^r = (\text{Sign}(dc_{ij}^s) * (\text{Abs}(dc_{ij}^s) * \text{LevelScale2}(QS_C \% 6, 0, 0) + (1 \ll (15 + QS_C / 6)))) \gg (16 + QS_C / 6) \quad \text{con } i, j = 0, 1 \quad (8-306)$$

Se calcula la matriz 2x2 f, de elementos  $f_{ij}$  con  $i, j = 0..1$ .

$$f = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} dc_{00}^r & dc_{01}^r \\ dc_{10}^r & dc_{11}^r \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}. \quad (8-307)$$

Se aplica un cambio de escala a los elementos  $f_{ij}$  de  $f$  siguiendo el procedimiento siguiente.

– Si  $QS_C$  es mayor o igual que 6, se calcula  $c_{00}()$  mediante la expresión:

$$c_{00}(j * 2 + i) = (f_{ij} * \text{LevelScale}(QS_C \% 6, 0, 0)) \ll (QS_C / 6 - 1) \text{ con } i, j = 0, 1 \quad (8-308)$$

– De lo contrario ( $QS_C$  es menor que 6), se calcula el  $c_{00}()$  mediante la expresión:

$$c_{00}(j * 2 + i) = (f_{ij} * \text{LevelScale}(QS_C \% 6, 0, 0)) \gg 1 \text{ con } i, j = 0, 1 \quad (8-309)$$

### 8.6.2 Proceso de decodificación de sectores SP y SI de imágenes intercambiadas

Se llama a este proceso al decodificar macrobloques de tipo P en sectores SP para los cuales `sp_for_switch_flag` es igual a 1, y al decodificar macrobloques de tipo SI en sectores SI.

Este proceso acepta como argumentos los niveles de coeficientes de transformada residuales de predicción y las matrices de muestras de predicción  $pred_L$ ,  $pred_{Cb}$ ,  $pred_{Cr}$  correspondientes al macrobloque considerado.

Este proceso genera como resultado las muestras decodificadas del macrobloque considerado antes de aplicar el proceso de filtrado de bloques.

#### 8.6.2.1 Proceso de decodificación de coeficientes de transformada luma

Este proceso acepta como argumentos las muestras luma de predicción  $pred_L$  y los niveles de coeficientes de transformada residuales de predicción luma, `LumaLevel`.

Este proceso genera como resultado las muestras luma decodificadas del macrobloque considerado antes de aplicar el proceso de filtrado de bloques.

Se calcula la matriz  $4 \times 4$   $p$ , de elementos  $p_{ij}$  con  $i, j = 0..3$ , que se describe en la subcláusula 8.6.1.1, y mediante la ecuación 8-290 se obtienen los coeficientes de transformada  $c^p$ . Estos coeficientes de transformada se cuantifican mediante el parámetro de cuantificación  $QS_Y$ , del modo siguiente:

$$c_{ij}^s = (\text{Sign}(c_{ij}^p) * (\text{Abs}(c_{ij}^p) * \text{LevelScale2}(QS_Y \% 6, i, j) + (1 \ll (14 + QS_Y / 6)))) \gg (15 + QS_Y / 6) \text{ con } i, j = 0..3 \quad (8-310)$$

Se llama al proceso de barrido inverso de coeficientes de transformada descrito en la subcláusula 8.5.4, pasándole como argumentos `LumaLevel[luma4x4BlkIdx]`, y se obtiene como resultado las dos matrices bidimensionales  $c^r$  de elementos  $c_{ij}^r$ .

La matriz  $4 \times 4$   $c$ , de elementos  $c_{ij}$  con  $i, j = 0..3$ , se calcula del modo siguiente:

$$c_{ij} = c_{ij}^r + c_{ij}^s \text{ con } i, j = 0..3 \quad (8-311)$$

Se llama al proceso de cambio de escala y transformación para bloques  $4 \times 4$  residuales descrito en la subcláusula 8.5.8, pasándole como argumento  $c$  y se obtiene como resultado  $r$ .

La matriz  $4 \times 4$   $u$  de elementos  $u_{ij}$  se calcula del modo siguiente:

$$u_{ij} = \text{Clip1}(r_{ij}) \text{ con } i, j = 0..3 \quad (8-312)$$

Se llama al proceso de reconstrucción de imágenes antes de aplicar el proceso de filtrado de bloques descrito en la subcláusula 8.5.9, pasándole como argumentos `luma4x4BlkIdx` y  $u$ , y el resultado es  $S'$ .

#### 8.6.2.2 Proceso de decodificación de coeficientes de transformada cromina

Este proceso acepta como argumentos muestras cromina predichas del macrobloque considerado, obtenidas según la subcláusula 8.4, y los niveles de coeficientes de transformada residuales de predicción `ChromaDCLevel` y `ChromaACLevel`.

Este proceso genera como resultado muestras cromas decodificadas del macrobloque considerado antes de aplicar el proceso de filtrado de bloques.

Se llama dos veces a este proceso: una vez para la componente Cb y otra para la componente Cr. Para hacer referencia a la componente Cb se sustituye C por Cb y para hacer referencia a la componente Cr se sustituye C por Cr. Sea iCbCr la componente cromas considerada.

A cada bloque 4x4 de la componente cromas considerada con índice chroma4x4BlkIdx, siendo chroma4x4BlkIdx igual a 0..3, se aplica lo siguiente.

- Se calcula la matriz 4x4 p, de elementos p<sub>ij</sub> con i, j = 0..3, descrita en la subcláusula 8.6.1.2, y se aplica la ecuación 8-290 para generar coeficientes de transformada c<sup>p</sup>( chroma4x4BlkIdx ). Estos coeficientes de transformada se cuantifican mediante el parámetro de cuantificación QS<sub>C</sub>, con i, j = 0..3 salvo para la combinación i = 0, j = 0. El cálculo de c<sub>00</sub><sup>p</sup>( chroma4x4BlkIdx ) se describe más adelante en esta subcláusula.

$$c_{ij}^s = ( \text{Sign}( c_{ij}^p( \text{chroma4x4BlkIdx} ) ) * ( \text{Abs}( c_{ij}^p( \text{chroma4x4BlkIdx} ) ) * \text{LevelScale2}( \text{QS}_C \% 6, i, j ) + ( 1 \ll ( 14 + \text{QS}_C / 6 ) ) ) ) \gg ( 15 + \text{QS}_C / 6 ) \quad (8-313)$$

- Se calcula la variable chromaList, que es una lista de 16 elementos. chromaList[ 0 ] se pone a 0. chromaList[ k ] con índice k = 1..15 se calcula del modo siguiente.

$$\text{chromaList}[ k ] = \text{ChromaACLevel}[ \text{iCbCr} ][ \text{chroma4x4BlkIdx} ][ k - 1 ] \quad (8-314)$$

- Se llama al proceso de barrido inverso de coeficiente de transformada descrito en la subcláusula 8.5.4, pasándole como argumento chromaList, y se obtiene como resultado las dos matrices bidimensional c<sup>r</sup>( chroma4x4BlkIdx ) de elementos c<sub>ij</sub><sup>r</sup>( chroma4x4BlkIdx ).
- Se calcula la matriz 4x4 c( chroma4x4BlkIdx ) de elementos c<sub>ij</sub>( chroma4x4BlkIdx ) con i, j = 0..3 salvo la combinación i = 0, j = 0. El cálculo de c<sub>00</sub>( chroma4x4BlkIdx ) se describe más adelante.

$$c_{ij}( \text{chroma4x4BlkIdx} ) = c_{ij}^r( \text{chroma4x4BlkIdx} ) + c_{ij}^s \quad (8-315)$$

- Se llama al proceso de cambio de escala y transformación para bloques 4x4 residuales, descrito en la subcláusula 8.5.8, pasándole como argumento c( chroma4x4BlkIdx ), y se obtiene como resultado r.

- Se calcula la matriz 4x4 u de elementos u<sub>ij</sub>.

$$u_{ij} = \text{Clip1}( r_{ij} ) \text{ con } i, j = 0..3 \quad (8-316)$$

- Se llama al proceso de reconstrucción de imágenes antes de aplicar el proceso de filtrado de bloques descrito en la subcláusula 8.5.9, pasándole como argumentos chroma4x4BlkIdx y u, y se obtiene como resultado S'.

A continuación se describe el cálculo del nivel de coeficiente de transformada c.c. c<sub>00</sub>( chroma4x4BlkIdx ). Los coeficientes de transformada c.c. de los 4 bloques cromas 4x4 de predicción de la componente considerada del macrobloque, c<sub>00</sub><sup>p</sup>( chroma4x4BlkIdx ), se agrupan en una matriz 2x2, y se aplica una transformada 2x2 a los coeficientes de transformada c.c. de estos bloques de acuerdo con la ecuación 8-304, obteniéndose así los coeficientes de transformada c.c. dc<sub>ij</sub><sup>p</sup>.

Seguidamente se cuantifican estos coeficientes de transformada c.c. mediante el parámetro de cuantificación QS<sub>C</sub>, utilizando la expresión:

$$dc_{ij}^s = ( \text{Sign}( dc_{ij}^p ) * ( \text{Abs}( dc_{ij}^p ) * \text{LevelScale2}( \text{QS}_C \% 6, 0, 0 ) + ( 1 \ll ( 15 + \text{QS}_C / 6 ) ) ) ) \gg ( 16 + \text{QS}_C / 6 ) \quad \text{con } i, j = 0, 1 \quad (8-317)$$

Los coeficientes de transformada residuales de predicción c.c. cromas resultantes, ChromaDCLevel[ iCbCr ][ k ] con k = 0..3, se suman a los coeficientes de transformada c.c. cuantificados del bloque de predicción, mediante la expresión:

$$dc_{ij}^r = dc_{ij}^s + \text{ChromaDCLevel}[ \text{iCbCr} ][ j * 2 + i ] \text{ con } i, j = 0, 1 \quad (8-318)$$

Se calcula la matriz 2x2  $f$ , de elementos  $f_{ij}$  con  $i, j = 0..1$ , mediante la ecuación 8-307.

El valor de los elementos  $f_{ij}$ , con  $i, j = 0..1$ , de la matriz 2x2  $f$  es:

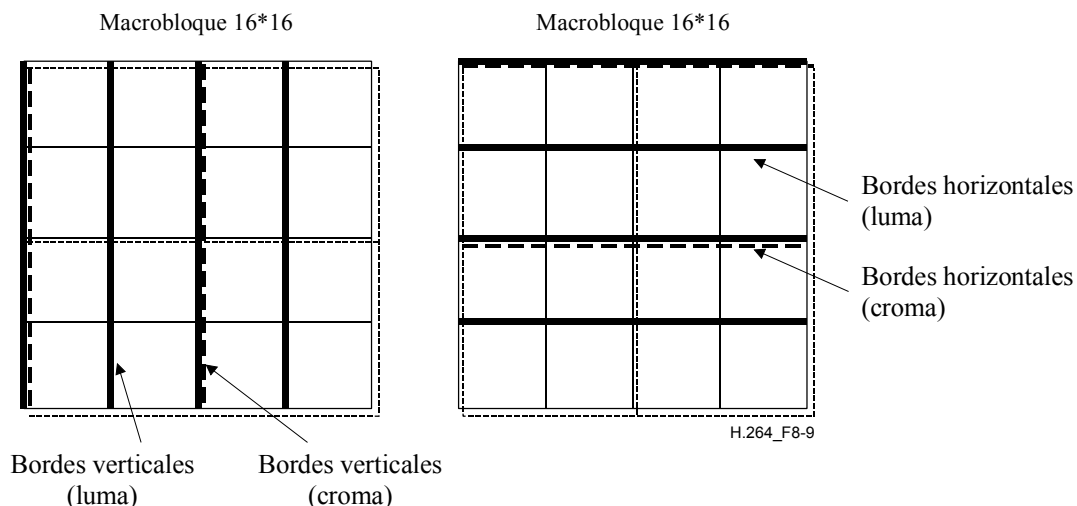
$$c_{00}(j * 2 + i) = f_{ij} \text{ con } i, j = 0, 1 \quad (8-319)$$

### 8.7 Proceso de filtrado de bloques

Se aplicará un filtrado condicional a los bordes de todos los bloques 4x4 de la imagen, excepto a los que coinciden con el contorno de la imagen y aquéllos para los cuales está deshabilitado el proceso de filtrado de bloques mediante `disable_deblocking_filter_idc`, según se describe a continuación. El proceso de filtrado se realiza por macrobloques, después de finalizar el proceso de construcción de la imagen y antes de realizar el proceso de filtrado de bloques (como se especifica en las subcláusulas 8.5 y 8.6) para la totalidad de la imagen decodificada, de modo que se procesan todos los macrobloques de la imagen en orden creciente de dirección de macrobloque.

NOTA – Antes de aplicar el proceso de filtrado de bloques a cada macrobloque, las muestras de los bloques ya filtrados del macrobloque o par de macrobloques anterior (si los hubiere) y del macrobloque o par de macrobloques a la izquierda (si los hubiere) del macrobloque considerado están siempre disponibles, debido a que el proceso de filtrado de bloques se lleva a cabo después de que finaliza el proceso de construcción de imagen y antes del proceso de filtrado de bloques para la imagen decodificada completa.

El proceso de filtrado de bloques se aplica por separado a las componentes luma y croma. Para cada macrobloque se filtran en primer lugar los bordes verticales, de izquierda a derecha, y después los bordes horizontales, de arriba abajo. El proceso de filtrado de bloques luma se aplica a cuatro bordes de 16 muestras, y el proceso de filtrado de bloques para cada componente croma se aplica a dos bordes de 8 muestras, como se muestra en la figura 8-9 de la izquierda para la dirección horizontal y en la figura 8-9 de la derecha para la dirección vertical. Al proceso de filtrado de bloques de un determinado macrobloque se le pasa como argumento los valores de las muestras que están por encima y a la izquierda del macrobloque considerado y a las que ya se les ha aplicado el proceso de filtrado de bloques a macrobloques anteriores; estos valores se pueden volver a modificar durante el filtrado del macrobloque considerado. Los valores de las muestras modificadas en el filtrado de los bordes verticales se utilizan para filtrar los bordes horizontales del mismo macrobloque.



**Figura 8-9 – Bordes de un macrobloque que se va a filtrar (los bordes luma se muestran con línea continua y los croma con líneas discontinuas)**

A cada macrobloque en orden ascendente de `mbAddr` se aplica lo siguiente.

1. Se calculan las variables `fieldModeMbFlag`, `filterInternalEdgesFlag`, `filterLeftMbEdgeFlag` y `filterTopMbEdgeFlag`.
  - Se calcula la variable `fieldModeMbFlag`.
    - Si se cumple algunas de las condiciones siguientes, `fieldModeMbFlag` se pone a 1.
      - `field_pic_flag` es igual a 1;
      - `MbaffFrameFlag` es igual a 1 y el macrobloque `mbAddr` es un macrobloque campo.
    - De lo contrario, `fieldModeMbFlag` se pone a 0.
  - Se calcula la variable `filterInternalEdgesFlag`.
    - Si `disable_deblocking_filter_idc` del sector que contiene el macrobloque `mbAddr` es igual a 1, la variable `filterInternalEdgesFlag` se pone a 0.
    - De lo contrario (`disable_deblocking_filter_idc` del sector que contiene el macrobloque `mbAddr` es distinto de 1), la variable `filterInternalEdgesFlag` se pone a 1.
  - Se calcula la variable `filterLeftMbEdgeFlag`.
    - Si se cumple alguna de las siguientes condiciones, la variable `filterLeftMbEdgeFlag` se pone a 0.
      - el borde vertical izquierdo del macrobloque `mbAddr` coincide con el contorno de la imagen;
      - `disable_deblocking_filter_idc` para el sector que contiene el macrobloque `mbAddr` es igual a 1;
      - `disable_deblocking_filter_idc` para el sector que contiene el macrobloque `mbAddr` es igual a 2, y el borde vertical izquierdo del macrobloque `mbAddr` coincide con el contorno de un sector.
    - De lo contrario, la variable `filterLeftMbEdgeFlag` se pone a 1.
  - Se calcula la variable `filterTopMbEdgeFlag`.
    - Si se cumple alguna de las siguientes condiciones, la variable `filterTopMbEdgeFlag` se pone a 0.
      - el borde horizontal superior del macrobloque `mbAddr` coincide con el contorno de la imagen;
      - `disable_deblocking_filter_idc` para el sector que contiene el macrobloque `mbAddr` es igual a 1;
      - `disable_deblocking_filter_idc` para el sector que contiene el macrobloque `mbAddr` es igual a 2, y el borde horizontal superior del macrobloque `mbAddr` coincide con el contorno de un sector.
    - De lo contrario, la variable `filterTopMbEdgeFlag` se pone a 1.
2. Dadas las variables `fieldModeMbFlag`, `filterInternalEdgesFlag`, `filterLeftMbEdgeFlag` y `filterTopMbEdgeFlag`, el filtrado de bloques se controla del modo siguiente.
  - Cuando `filterLeftMbEdgeFlag` es igual a 1, el filtrado del borde luma vertical izquierdo se define del modo siguiente.
    - Se llama al proceso descrito en la subcláusula 8.7.1, pasándole como argumentos `mbAddr`, `chromaEdgeFlag = 0`, `verticalEdgeFlag = 1`, `fieldModeFilteringFlag = fieldModeMbFlag`, y  $(xE_k, yE_k) = (0, k)$ , con  $k = 0..15$ , y el resultado es  $S'_L$ .
  - Cuando `filterInternalEdgesFlag` es igual a 1, el filtrado de los bordes luma verticales internos se define del modo siguiente.

- Se llama al proceso descrito en la subcláusula 8.7.1, pasándole como argumentos  $mbAddr$ ,  $chromaEdgeFlag = 0$ ,  $verticalEdgeFlag = 1$ ,  $fieldModeFilteringFlag = fieldModeMbFlag$ , y  $(xE_k, yE_k) = (4, k)$ , con  $k = 0..15$ , y el resultado es  $S'_L$ .
- Se llama al proceso descrito en la subcláusula 8.7.1, pasándole como argumentos  $mbAddr$ ,  $chromaEdgeFlag = 0$ ,  $verticalEdgeFlag = 1$ ,  $fieldModeFilteringFlag = fieldModeMbFlag$ , y  $(xE_k, yE_k) = (8, k)$ , con  $k = 0..15$ , y el resultado es  $S'_L$ .
- Se llama al proceso descrito en la subcláusula 8.7.1, pasándole como argumentos  $mbAddr$ ,  $chromaEdgeFlag = 0$ ,  $verticalEdgeFlag = 1$ ,  $fieldModeFilteringFlag = fieldModeMbFlag$ , y  $(xE_k, yE_k) = (12, k)$ , con  $k = 0..15$ , y el resultado es  $S'_L$ .
- Cuando  $filterTopMbEdgeFlag$  es igual a 1, el filtrado del borde luma horizontal superior se define del modo siguiente.
  - Si  $MbaffFrameFlag$  es igual a 1,  $(mbAddr \% 2)$  es igual a 0,  $mbAddr$  es mayor o igual a  $2 * PicWidthInMbs$ , el macrobloque  $mbAddr$  es un macrobloque cuadro y el macrobloque  $(mbAddr - 2 * PicWidthInMbs + 1)$  es un macrobloque campo, se aplica lo siguiente.
    - Se llama al proceso descrito en la subcláusula 8.7.1, pasándole como argumentos  $mbAddr$ ,  $chromaEdgeFlag = 0$ ,  $verticalEdgeFlag = 0$ ,  $fieldModeFilteringFlag = 1$ , y  $(xE_k, yE_k) = (k, 0)$ , con  $k = 0..15$ , y el resultado es  $S'_L$ .
    - Se llama al proceso descrito en la subcláusula 8.7.1, pasándole como argumentos  $mbAddr$ ,  $chromaEdgeFlag = 0$ ,  $verticalEdgeFlag = 0$ ,  $fieldModeFilteringFlag = 1$ , y  $(xE_k, yE_k) = (k, 1)$ , con  $k = 0..15$ , y el resultado es  $S'_L$ .
  - De lo contrario, se llama al proceso descrito en la subcláusula 8.7.1, pasándole como argumentos  $mbAddr$ ,  $chromaEdgeFlag = 0$ ,  $verticalEdgeFlag = 0$ ,  $fieldModeFilteringFlag = fieldModeMbFlag$ , y  $(xE_k, yE_k) = (k, 0)$ , con  $k = 0..15$ , y el resultado es  $S'_L$ .
- Cuando  $filterInternalEdgesFlag$  es igual a 1, el filtrado de los bordes luma horizontales internos se define del modo siguiente.
  - Se llama al proceso descrito en la subcláusula 8.7.1, pasándole como argumentos  $mbAddr$ ,  $chromaEdgeFlag = 0$ ,  $verticalEdgeFlag = 0$ ,  $fieldModeFilteringFlag = fieldModeMbFlag$ , y  $(xE_k, yE_k) = (k, 4)$ , con  $k = 0..15$ , y el resultado es  $S'_L$ .
  - Se llama al proceso descrito en la subcláusula 8.7.1, pasándole como argumentos  $mbAddr$ ,  $chromaEdgeFlag = 0$ ,  $verticalEdgeFlag = 0$ ,  $fieldModeFilteringFlag = fieldModeMbFlag$ , y  $(xE_k, yE_k) = (k, 8)$ , con  $k = 0..15$ , y el resultado es  $S'_L$ .
  - Se llama al proceso descrito en la subcláusula 8.7.1, pasándole como argumentos  $mbAddr$ ,  $chromaEdgeFlag = 0$ ,  $verticalEdgeFlag = 0$ ,  $fieldModeFilteringFlag = fieldModeMbFlag$ , y  $(xE_k, yE_k) = (k, 12)$ , con  $k = 0..15$ , y el resultado es  $S'_L$ .
- A las dos componentes croma,  $iCbCr = 0$  y 1, se aplica lo siguiente.
  - Cuando  $filterLeftMbEdgeFlag$  es igual a 1, el filtrado del borde croma vertical izquierdo se define del modo siguiente.
    - Se llama al proceso descrito en la subcláusula 8.7.1, pasándole como argumentos  $mbAddr$ ,  $chromaEdgeFlag = 1$ ,  $iCbCr$ ,  $verticalEdgeFlag = 1$ ,  $fieldModeFilteringFlag = 1$ , y  $(xE_k, yE_k) = (0, k)$ , con  $k = 0..7$ , y el resultado es  $S'_C$ , siendo  $C$  igual a  $Cb$  para  $iCbCr = 0$ , y  $C$  igual a  $Cr$  para  $iCbCr = 1$ .
  - Cuando  $filterInternalEdgesFlag$  es igual a 1, el filtrado del borde croma vertical interno se define del modo siguiente.



- Se llama al proceso descrito en la subcláusula 8.7.1, pasándole como argumentos  $mbAddr$ ,  $chromaEdgeFlag = 1$ ,  $iCbCr$ ,  $verticalEdgeFlag = 1$ ,  $fieldModeFilteringFlag = fieldModeMbFlag$ , y  $(xE_k, yE_k) = (4, k)$ , con  $k = 0..7$ , y el resultado es  $S'_C$ , siendo  $C$  igual a  $C_b$  para  $iCbCr = 0$ , y  $C$  igual a  $C_r$  para  $iCbCr = 1$ .
- Cuando  $filterTopMbEdgeFlag$  es igual a 1, el filtrado del borde croma horizontal superior se define del modo siguiente.
  - Si  $MbaffFrameFlag$  es igual a 1,  $(mbAddr \% 2)$  es igual a 0,  $mbAddr$  es mayor o igual que  $2 * PicWidthInMbs$ , el macrobloque  $mbAddr$  es un macrobloque cuadro y el macrobloque  $(mbAddr - 2 * PicWidthInMbs + 1)$  es un macrobloque campo, se aplica lo siguiente.
    - Se llama al proceso descrito en la subcláusula 8.7.1, pasándole como argumentos  $mbAddr$ ,  $chromaEdgeFlag = 1$ ,  $iCbCr$ ,  $verticalEdgeFlag = 0$ ,  $fieldModeFilteringFlag = 1$ , y  $(xE_k, yE_k) = (k, 0)$ , con  $k = 0..7$ , y el resultado es  $S'_C$ , siendo  $C$  igual a  $C_b$  para  $iCbCr = 0$  e igual a  $C_r$  para  $iCbCr = 1$ .
    - Se llama al proceso descrito en la subcláusula 8.7.1, pasándole como argumentos  $mbAddr$ ,  $chromaEdgeFlag = 1$ ,  $iCbCr$ ,  $verticalEdgeFlag = 0$ ,  $fieldModeFilteringFlag = 1$ , y  $(xE_k, yE_k) = (k, 1)$ , con  $k = 0..7$ , y el resultado es  $S'_C$ , siendo  $C$  igual a  $C_b$  para  $iCbCr = 0$  e igual a  $C_r$  para  $iCbCr = 1$ .
  - De lo contrario, se llama al proceso descrito en la subcláusula 8.7.1, pasándole como argumentos  $mbAddr$ ,  $chromaEdgeFlag = 1$ ,  $iCbCr$ ,  $verticalEdgeFlag = 0$ ,  $fieldModeFilteringFlag = fieldModeMbFlag$ , y  $(xE_k, yE_k) = (k, 0)$ , con  $k = 0..7$ , y el resultado es  $S'_C$ , siendo  $C$  igual a  $C_b$  para  $iCbCr = 0$  e igual a  $C_r$  para  $iCbCr = 1$ .
- Cuando  $filterInternalEdgesFlag$  es igual a 1, el filtrado del borde croma horizontal interno se define del modo siguiente.
  - Se llama al proceso descrito en la subcláusula 8.7.1, pasándole como argumentos  $mbAddr$ ,  $chromaEdgeFlag = 1$ ,  $iCbCr$ ,  $verticalEdgeFlag = 0$ ,  $fieldModeFilteringFlag = fieldModeMbFlag$ , y  $(xE_k, yE_k) = (k, 4)$ , con  $k = 0..7$ , y el resultado es  $S'_C$ , siendo  $C$  igual a  $C_b$  para  $iCbCr = 0$  e igual a  $C_r$  para  $iCbCr = 1$ .

NOTA – Cuando se aplica el filtrado en modo campo ( $fieldModeFilteringFlag$  es igual a 1) a los bordes horizontales superiores de un macrobloque cuadro, el filtrado vertical del contorno del macrobloque superior o inferior puede implicar que también se filtren internamente en modo cuadro algunas muestras que están más allá del borde interno.

NOTA – En todos los casos se filtran 3 bordes luma horizontales, 1 borde croma horizontal para  $C_b$ , y 1 borde croma horizontal para  $C_r$  que son internos a un macrobloque. Cuando se aplica el filtrado en modo campo ( $fieldModeFilteringFlag$  es igual a 1) a los bordes superiores de un macrobloque cuadro, se filtran en modo campo 2 bordes luma horizontales, 2 bordes croma horizontales para  $C_b$ , y 2 bordes croma horizontales para  $C_r$ , entre el macrobloque cuadro y el par de macrobloques anterior, lo que da un total de 5 bordes luma horizontales, 3 bordes croma horizontales para  $C_b$  y 3 bordes croma horizontales para  $C_r$  que se consideran controlados por el macrobloque cuadro. En todos los demás casos se filtran como máximo 4 bordes luma horizontales, 2 bordes croma horizontales para  $C_b$  y 2 bordes croma horizontales para  $C_r$ , que se consideran controlados por un determinado macrobloque.

Por último, las matrices  $S'_L$ ,  $S'_{Cb}$ ,  $S'_{Cr}$  se asignan, respectivamente, a las matrices  $S_L$ ,  $S_{Cb}$ ,  $S_{Cr}$  (que representan imagen decodificada).

### 8.7.1 Proceso de filtrado de bordes de bloque

Este proceso acepta como argumentos mbAddr, chromaEdgeFlag, el índice de la componente croma iCbCr (cuando chromaEdgeFlag es igual a 1), verticalEdgeFlag, fieldModeFilteringFlag, y un conjunto de 16 posiciones de muestras luma (cuando chromaEdgeFlag es igual a 0) u ocho posiciones de muestras croma (cuando chromaEdgeFlag es igual a 1) ( $xE_k, yE_k$ ), con  $k = 0 \dots nE - 1$ , que son posiciones relativas a la esquina superior izquierda del macrobloque mbAddr. El conjunto de posiciones de muestras ( $xE_k, yE_k$ ) representa las posiciones de la muestra exactamente a la derecha del borde vertical (cuando verticalEdgeFlag es igual a 1) o exactamente por debajo del borde horizontal (cuando verticalEdgeFlag es igual a 0).

La variable nE se calcula del modo siguiente.

- Si chromaEdgeFlag es igual a 0, nE es igual a 16;
- De lo contrario (chromaEdgeFlag es igual a 1), nE es igual a 8.

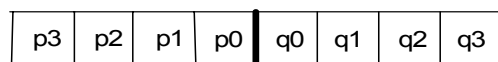
Se calcula la variable s' que especifica la matriz de muestras luma o croma.

- Si chromaEdgeFlag es igual a 0, s' representa la matriz de muestras luma  $S'_L$  de la imagen considerada.
- De lo contrario, si chromaEdgeFlag es igual a 1 y iCbCr es igual a 0, s' representa la matriz de muestras croma  $S'_{Cb}$  de la componente croma Cb de la imagen considerada.
- De lo contrario (chromaEdgeFlag es igual a 1 y iCbCr es igual a 1), s' representa la matriz de muestras croma  $S'_{Cr}$  de la componente croma Cr de la imagen considerada.

Se calcula la variable dy.

- Si fieldModeFilteringFlag es igual a 1 y MbaffFrameFlag es igual a 1, dy se pone a 2.
- De lo contrario (fieldModeFilteringFlag es igual a 0 o MbaffFrameFlag es igual a 0), dy se pone a 1.

Para calcular la posición de la muestra luma superior izquierda del macrobloque mbAddr se llama al proceso de barrido inverso de macrobloques descrito en la subcláusula 6.4.1, al que se le pasa como argumento mbAddr y el resultado se asigna a ( $xP, yP$ ).



**Figura 8-10 – Convenio para describir muestras que están en el contorno horizontal o vertical de un bloque 4x4**

A cada posición de muestra ( $xE_k, yE_k$ ),  $k = 0 \dots nE - 1$ , se aplica lo siguiente.

- Se aplica el proceso de filtrado a un conjunto de ocho muestras que están en el borde horizontal o vertical del bloque 4x4, indicadas mediante  $p_i$  y  $q_i$ , con  $i = 0..3$  como se muestra en la figura 8-10 de modo que el borde esté entre  $p_0$  y  $q_0$ . Las muestras  $p_i$  y  $q_i$  con  $i = 0..3$  se definen del modo siguiente.

- Si verticalEdgeFlag es igual a 1,

$$q_i = s'[ xP + xE_k + i, yP + yE_k ] \quad (8-320)$$

$$p_i = s'[ xP + xE_k - i - 1, yP + yE_k ] \quad (8-321)$$

- De lo contrario (verticalEdgeFlag es igual a 0),

$$q_i = s'[ xP + xE_k, yP + dy * ( yE_k + i ) - ( yE_k \% 2 ) ] \quad (8-322)$$

$$p_i = s'[ xP + xE_k, yP + dy * ( yE_k - i - 1 ) - ( yE_k \% 2 ) ] \quad (8-323)$$

- Se llama al proceso descrito en la subcláusula 8.7.2, pasándole como argumentos los valores de muestras  $p_i$  y  $q_i$  ( $i = 0..3$ ), chromaEdgeFlag, verticalEdgeFlag, y

fieldModeFilteringFlag, y el resultado se asigna a los valores de las muestras filtradas  $p'_i$  y  $q'_i$ , con  $i = 0..2$ .

- Se sustituyen los valores de las muestras  $p_i$  y  $q_i$ , con  $i = 0..2$ , por los correspondientes valores de muestras filtradas  $p'_i$  y  $q'_i$ , con  $i = 0..2$ , en la matriz de muestras siguiendo el procedimiento siguiente.

- Si verticalEdgeFlag es igual a 1,

$$s'[xP + xE_k + i, yP + yE_k] = q'_i \quad (8-324)$$

$$s'[xP + xE_k - i - 1, yP + yE_k] = p'_i \quad (8-325)$$

- De lo contrario (verticalEdgeFlag es igual a 0),

$$s'[xP + xE_k, yP + dy * (yE_k + i) - (yE_k \% 2)] = q'_i \quad (8-326)$$

$$s'[xP + xE_k, yP + dy * (yE_k - i - 1) - (yE_k \% 2)] = p'_i \quad (8-327)$$

### 8.7.2 Proceso de filtrado para un conjunto de muestras que se encuentran en el borde horizontal o vertical del bloque

Este proceso acepta como argumentos los valores de las muestras  $p_i$  y  $q_i$ , con  $i$  igual a  $0..3$ , de un conjunto de muestras que están en el borde que se va a filtrar, chromaEdgeFlag, verticalEdgeFlag y fieldModeFilteringFlag.

Este proceso genera como resultado los valores de las muestras filtradas  $p'_i$  y  $q'_i$ , con  $i$  igual a  $0..2$ .

Se calcula la variable bS de intensidad de filtrado del contorno dependiente del contenido.

- Si chromaEdgeFlag es igual a 0 se llama al proceso de cálculo de la intensidad de filtrado del contorno dependiente del contenido descrito en la subcláusula 8.7.2.1, al que se le pasa como parámetros  $p_0$ ,  $q_0$  y verticalEdgeFlag y el resultado se asigna a bS.
- De lo contrario (chromaEdgeFlag es igual a 1) se aplica lo siguiente.
  - Si fieldModeFilteringFlag es igual a 0, el bS utilizado para filtrar el conjunto de muestras de un borde croma horizontal o vertical será igual al valor de bS correspondiente al filtrado de un conjunto de muestras de un borde luma horizontal o vertical, respectivamente, que contiene la muestra luma en la posición  $(2 * x, 2 * y)$  dentro de la matriz luma del cuadro, donde  $(x, y)$  es la posición de la muestra croma  $q_0$  dentro de la matriz croma para ese cuadro.
  - De lo contrario (fieldModeFilteringFlag es igual a 1), el bS utilizado para filtrar el conjunto de muestras de un borde croma horizontal o vertical será igual al valor de bS correspondiente al filtrado de un conjunto de muestras de un borde luma horizontal o vertical, respectivamente, que contiene la muestra luma en la posición  $(2 * x, 2 * y)$  dentro de la matriz luma del cuadro, donde  $(x, y)$  es la posición de la muestra croma  $q_0$  dentro de la matriz croma para ese campo.

Se llama al proceso descrito en la subcláusula 8.7.2.2, pasándole como argumentos  $p_0$ ,  $q_0$ ,  $p_1$ ,  $q_1$ , chromaEdgeFlag, y bS y el resultado se asigna a filterSamplesFlag, indexA,  $\alpha$ , y  $\beta$ .

En función de la variable filterSamplesFlag, se aplica lo siguiente.

- Si filterSamplesFlag es igual a 1, se aplica lo siguiente.
  - Si bS es menor que 4, se llama al proceso descrito en la subcláusula 8.7.2.3, pasándole como argumentos  $p_i$  y  $q_i$  ( $i = 0..3$ ), chromaEdgeFlag, bS,  $\beta$  y indexA, y el resultado se asigna a  $p'_i$  y  $q'_i$  ( $i = 0..2$ ).
  - De lo contrario (bS es igual a 4), se llama al proceso descrito en la subcláusula 8.7.2.4, pasándole como argumentos  $p_i$  y  $q_i$  ( $i = 0..3$ ), chromaEdgeFlag,  $\alpha$  y  $\beta$ , y el resultado se asigna a  $p'_i$  y  $q'_i$  ( $i = 0..2$ ).

- De lo contrario (`filterSamplesFlag` es igual a 0), el valor de las muestras filtradas  $p'_i$  y  $q'_i$  ( $i = 0..2$ ) será igual al de las correspondientes muestras sin filtrar  $p_i$  y  $q_i$ :

$$\text{para } i = 0..2, \quad p'_i = p_i \quad (8-328)$$

$$\text{para } i = 0..2, \quad q'_i = q_i \quad (8-329)$$

### 8.7.2.1 Proceso de cálculo de la intensidad de filtrado del contorno dependiente del contenido luma

Este proceso acepta como argumentos los valores de muestra  $p_0$  y  $q_0$  de un solo conjunto de muestras situadas sobre el borde que se desea filtrar y `verticalEdgeFlag`.

Este proceso genera como resultado la variable `bS`.

Se calcula la variable `mixedModeEdgeFlag`.

- Si `MbaffFrameFlag` es igual a 1 y las muestras  $p_0$  y  $q_0$  están en pares de macrobloques diferente, uno de los cuales es un par de macrobloques campo y el otro es un par de macrobloques cuadro, `mixedModeEdgeFlag` se pone a 1.
- De lo contrario, `mixedModeEdgeFlag` se pone a 0.

Se calcula la variable `bS`.

- Si el borde del bloque es también un borde de macrobloque y se cumple alguna de las siguientes condiciones, el valor de `bS` será 4:
  - la muestra  $p_0$  y  $q_0$  están ambas en macrobloques cuadro y la muestra  $p_0$  o la  $q_0$  o ambas están en un macrobloque codificado mediante el modo intra de predicción de macrobloque.
  - las muestras  $p_0$  y  $q_0$  están ambas en macrobloques cuadro y una o las dos muestras  $p_0$  o  $q_0$  están en un macrobloque que es un sector cuyo `slice_type` es igual a SP o SI.
  - `MbaffFrameFlag` es igual a 1 o `field_pic_flag` es igual a 1, y `verticalEdgeFlag` es igual a 1 y la muestra  $p_0$  o la  $q_0$  o ambas están en un macrobloque codificado mediante el modo intra predicción de macrobloque.
  - `MbaffFrameFlag` es igual a 1 o `field_pic_flag` es igual a 1, `verticalEdgeFlag` es igual a 1, y una o ambas de las muestras  $p_0$  o  $q_0$  están en un macrobloque que es un sector cuyo `slice_type` es igual a SP o a SI.
- De lo contrario, si se cumple alguna de las siguientes condiciones, el valor de `bS` será igual a 3:
  - `mixedModeEdgeFlag` es igual a 0 y la muestra  $p_0$  o la  $q_0$  o ambas están en un macrobloque codificado mediante el modo Intra de predicción de macrobloque.
  - `mixedModeEdgeFlag` es igual a 0 y una o ambas de las muestras  $p_0$  o  $q_0$  están en un macrobloque que se encuentra en un sector cuyo `slice_type` es igual a SP o a SI.
  - `mixedModeEdgeFlag` es igual a 1, `verticalEdgeFlag` es igual a 0 y la muestra  $p_0$  o la  $q_0$  o ambas están en un macrobloque codificado mediante el modo intra de predicción de macrobloque.
  - `mixedModeEdgeFlag` es igual a 1, `verticalEdgeFlag` es igual a 0, y una o ambas de las muestras  $p_0$  y  $q_0$  se encuentran en un macrobloque que está en un sector cuyo `slice_type` es igual a SP o a SI.
- De lo contrario, si se cumple la siguiente condición, el valor de `bS` será igual a 2:
  - el bloque luma 4x4 que contiene la muestra  $p_0$  o el bloque luma 4x4 que contiene la muestra  $q_0$  contienen niveles de coeficientes de transformada distintos de cero.
- De lo contrario, si se cumple alguna de las siguientes condiciones, el valor de `bS` será igual a 1:

- mixedModeEdgeFlag es igual a 1.
- mixedModeEdgeFlag es igual a 0 y para predecir la partición macrobloque/submacrobloque que contiene la muestra  $p_0$  se utilizan imágenes de referencia diferentes o un número diferente de vectores de movimiento que los utilizados para predecir la partición macrobloque/submacrobloque que contiene la muestra  $q_0$ .

NOTA – La determinación de si las imágenes de referencia utilizadas para las dos particiones de macrobloque/submacrobloque son la misma o diferentes se basa sólo en las imágenes referenciadas, sin importar si se forma una predicción mediante la utilización de un índice en la lista 0 o de un índice en la lista 1, y sin que importe tampoco si es diferente o no la posición del índice en una lista de imágenes de referencia.

- mixedModeEdgeFlag es igual a 0 y se utiliza un vector de movimiento para predecir la partición macrobloque/submacrobloque que contiene la muestra  $p_0$  y se utiliza un vector de movimiento para predecir la partición macrobloque/submacrobloque que contiene la muestra  $q_0$ , y la diferencia en valor absoluto entre la componente horizontal y la vertical del vector del movimiento utilizado es mayor o igual que 4 en unidades de un cuarto de muestra cuadro luma.
- mixedModeEdgeFlag es igual a 0 y se utilizan dos vectores de movimiento y dos imágenes de referencia diferentes para predecir la partición macrobloque/submacrobloque que contiene la muestra  $p_0$  y se utilizan dos vectores de movimiento de las mismas dos imágenes de referencia para predecir la partición macrobloque/submacrobloque que contiene la muestra  $q_0$ , y la diferencia en valor absoluto entre la componente horizontal o la vertical de un vector de movimiento utilizado para predecir las dos particiones macrobloque/submacrobloque para la misma imagen de referencia es mayor o igual a 4 en unidades de un cuarto de muestra cuadro luma.
- mixedModeEdgeFlag es igual a 0 y se utilizan dos vectores de movimiento de la misma imagen de referencia para predecir la partición macrobloque/submacrobloque que contiene la muestra  $p_0$  y se utilizan dos vectores de movimiento de la misma imagen de referencia para predecir la partición macrobloque/submacrobloque que contiene la muestra  $q_0$  y se cumplen las dos condiciones siguientes:
  - La diferencia en valor absoluto entre la componente horizontal o vertical de los vectores de movimiento de la lista 0 utilizados en la predicción de las dos particiones macrobloque/submacrobloque es mayor o igual a 4 en unidades de cuarto de muestra cuadro luma, o la diferencia en valor absoluto entre la componente horizontal o vertical de los vectores de movimiento de lista 1 utilizados en la predicción en las dos particiones macrobloque/submacrobloque es mayor o igual a 4 unidades de un cuarto de muestra cuadro luma.
  - La diferencia en valor absoluto entre la componente horizontal o vertical del vector de movimiento de lista 0 utilizado en la predicción de la partición macrobloque/submacrobloque que contiene la muestra  $p_0$  y el vector de movimiento de lista 1 utilizado en la predicción de la partición macrobloque/submacrobloque que contiene la muestra  $q_0$  es mayor o igual a 4 en unidades de un cuarto de muestra cuadro luma, o la diferencia en valor absoluto entre la componente horizontal o vertical del vector de movimiento de lista 1 utilizado en la predicción de la partición de macrobloque/submacrobloque que contiene la muestra  $p_0$  y el vector de movimiento de lista 0 utilizado en la predicción de la partición macrobloque/submacrobloque que contiene la muestra  $q_0$  es mayor o igual a 4 en unidades de un cuarto de muestra cuadro luma.

NOTA – Una diferencia de 4 unidades de un cuarto de muestra de cuadro luma en la componente vertical equivale a una diferencia de 2 en unidades de cuarto de muestra campo luma.

- De lo contrario, el valor de bS será igual a 0.

### 8.7.2.2 Proceso de cálculo de los umbrales para cada borde del bloque

Este proceso acepta como argumentos los valores de las muestras  $p_0$ ,  $q_0$ ,  $p_1$  y  $q_1$  de un solo conjunto de muestras situadas en el borde que se desea filtrar, y las variables `chromaEdgeFlag` y `bS` del conjunto de muestras que se pasan como argumento, según se especifica en 8.7.2.

Este proceso genera como resultado la variable `filterSamplesFlag` que indica si las muestras están filtradas, el valor de `indexA` y los valores de las variables umbral  $\alpha$  y  $\beta$ .

Sean  $qP_p$  y  $qP_q$  variables que especifican los valores del parámetro de cuantificación de los macrobloques que contienen las muestras  $p_0$  y  $q_0$ , respectivamente. Las variables  $qP_z$  (siendo  $z$   $p$  o  $q$ ) se calculan del modo siguiente.

- Si `chromaEdgeFlag` es igual a 0, se aplica lo siguiente.
  - Si el macrobloque que contiene la muestra  $z_0$  es un macrobloque I\_PCM,  $qP_z$  se pone a 0.
  - De lo contrario (el macrobloque que contiene la muestra  $z_0$  no es un macrobloque I\_PCM),  $qP_z$  se pone al valor de  $QP_Y$  del macrobloque que contiene la muestra  $z_0$ .
- De lo contrario (`chromaEdgeFlag` es igual a 1), se aplica lo siguiente.
  - Si el macrobloque que contiene la muestra  $z_0$  es un macrobloque I\_PCM,  $qP_z$  se pone al valor de  $QP_C$  que corresponde a un valor de 0 para  $QP_Y$ , como se describe en la subcláusula 8.5.5.
  - De lo contrario (el macrobloque que contiene la muestra  $z_0$  no es un macrobloque I\_PCM),  $qP_z$  se pone al valor de  $QP_C$  que corresponde al valor de  $QP_Y$  del macrobloque que contiene la muestra  $z_0$ , como se describe en la subcláusula 8.5.5.

Se calcula la variable  $qP_{av}$  que especifica un parámetro de cuantificación media.

$$qP_{av} = (qP_p + qP_q + 1) \gg 1 \quad (8-330)$$

NOTA – En sectores SP y SI,  $qP_{av}$  se calcula del mismo modo que en otros tipos de sector.  $QS_Y$  de la ecuación 7-17 no se utiliza en el filtrado de bloques.

Sea `indexA` una variable que se utiliza para acceder al cuadro de  $\alpha$  (cuadro 8-14) y al cuadro de  $t_{C0}$  (cuadro 8-15) las cuales se utilizan para filtrar los bordes cuando `bS` es menor que 4, según se describe en la subcláusula 8.7.2.3, y sea `indexB` la variable que se utiliza para acceder al cuadro de  $\beta$  (cuadro 8-14). Se calcula el valor de las variables `indexA` e `indexB`, donde los valores de `FilterOffsetA` y `FilterOffsetB` descritos en la subcláusula 7.4.3 son los valores de esas variables del sector que contiene el macrobloque que a su vez contiene la muestra  $q_0$ .

$$\text{indexA} = \text{Clip3}(0, 51, qP_{av} + \text{FilterOffsetA}) \quad (8-331)$$

$$\text{indexB} = \text{Clip3}(0, 51, qP_{av} + \text{FilterOffsetB}) \quad (8-332)$$

En el cuadro 8-14 se indican las variables umbral  $\alpha$  y  $\beta$  en función de los valores de `indexA` e `indexB`.

Se calcula la variable `filterSamplesFlag`

$$\text{filterSamplesFlag} = (\text{bS} \neq 0 \ \&\& \ \text{Abs}(p_0 - q_0) < \alpha \ \&\& \ \text{Abs}(p_1 - p_0) < \beta \ \&\& \ \text{Abs}(q_1 - q_0) < \beta) \quad (8-333)$$

**Cuadro 8-14 – Cálculo de indexA e indexB a partir de las variables umbral  $\alpha$  y  $\beta$  dependientes de la traslación**

		indexA (para $\alpha$ ) o indexB (para $\beta$ )																									
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
$\alpha$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	4	5	6	7	8	9	10	12	13
$\beta$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	2	3	3	3	3	4	4	4	

**Cuadro 8-14 (continuación) – Cálculo de indexA e indexB a partir de las variables umbral  $\alpha$  y  $\beta$  dependientes de la traslación**

		indexA (para $\alpha$ ) o indexB (para $\beta$ )																													
		26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51				
$\alpha$	15	17	20	22	25	28	32	36	40	45	50	56	63	71	80	90	101	113	127	144	162	182	203	226	255	255					
$\beta$	6	6	7	7	8	8	9	9	10	10	11	11	12	12	13	13	14	14	15	15	16	16	17	17	18	18					

### 8.7.2.3 Proceso de filtrado de bordes con bS menor que 4

Ese proceso acepta como argumentos los valores de muestra  $p_i$  y  $q_i$  ( $i = 0..2$ ) de un solo conjunto de muestras situadas en un borde que se va a filtrar, y las variables `chromaEdgeFlag`, `bS`,  $\beta$ , e `indexA`, del conjunto de muestras que se pasan como argumentos, según se especifica en 8.7.2.

Este proceso genera como resultado los valores de las muestras filtradas  $p'_i$  y  $q'_i$  ( $i = 0..2$ ) para el conjunto de valores de muestras.

Las muestras filtradas  $p'_0$  y  $q'_0$  se calculan mediante las expresiones

$$\Delta = \text{Clip3}(-t_c, t_c, (((q_0 - p_0) \ll 2) + (p_1 - q_1) + 4) \gg 3)) \quad (8-334)$$

$$p'_0 = \text{Clip1}(p_0 + \Delta) \quad (8-335)$$

$$q'_0 = \text{Clip1}(q_0 - \Delta) \quad (8-336)$$

donde el umbral  $t_c$  se determina del modo siguiente.

– Si `chromaEdgeFlag` es igual a 0,

$$t_c = t_{c0} + ((a_p < \beta) ? 1 : 0) + ((a_q < \beta) ? 1 : 0) \quad (8-337)$$

– De lo contrario (`chromaEdgeFlag` es igual a 1),

$$t_c = t_{c0} + 1 \quad (8-338)$$

En el cuadro 8-15 se indica el umbral  $t_{c0}$  en función de los valores de `indexA` y `bS`.

Sean  $a_p$  y  $a_q$  las dos variables umbral especificadas mediante

$$a_p = \text{Abs}(p_2 - p_0) \quad (8-339)$$

$$a_q = \text{Abs}(q_2 - q_0) \quad (8-340)$$

La muestra filtrada  $p'_1$  se calcula del modo siguiente

– Si `chromaEdgeFlag` es igual a 0 y  $a_p$  es menor que  $\beta$ ,

$$p'_1 = p_1 + \text{Clip3}(-t_{c0}, t_{c0}, (p_2 + ((p_0 + q_0 + 1) \gg 1) - (p_1 \ll 1)) \gg 1) \quad (8-341)$$

– De lo contrario (`chromaEdgeFlag` es igual a 1 o  $a_p$  es mayor o igual que  $\beta$ ),

$$p'_1 = p_1 \quad (8-342)$$

La muestra filtrada  $q'_1$  se calcula del modo siguiente

– Si `chromaEdgeFlag` es igual a 0 y  $a_q$  es menor que  $\beta$ ,

$$q'_1 = q_1 + \text{Clip3}(-t_{c0}, t_{c0}, (q_2 + ((p_0 + q_0 + 1) \gg 1) - (q_1 \ll 1)) \gg 1) \quad (8-343)$$

– De lo contrario (chromaEdgeFlag es igual a 1 o  $a_q$  es mayor o igual que  $\beta$ ),

$$q'_1 = q_1 \quad (8-344)$$

A las muestras filtradas  $p'_2$  y  $q'_2$  siempre se les asigna el mismo valor que las muestras  $p_2$  y  $q_2$ :

$$p'_2 = p_2 \quad (8-345)$$

$$q'_2 = q_2 \quad (8-346)$$

**Cuadro 8-15 – Valor de la variable de recorte del filtro  $t_{c0}$  en función de indexA y bS**

	indexA																									
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
bS = 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
bS = 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
bS = 3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1

**Cuadro 8-15 (continuación) – Valor de la variable de recorte del filtro  $t_{c0}$  en función de indexA y bS**

	indexA																													
	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51				
bS = 1	1	1	1	1	1	1	1	2	2	2	2	3	3	3	4	4	4	5	6	6	7	8	9	10	11	13				
bS = 2	1	1	1	1	1	2	2	2	2	3	3	3	4	4	5	5	6	7	8	8	10	11	12	13	15	17				
bS = 3	1	2	2	2	2	3	3	3	4	4	4	5	6	6	7	8	9	10	11	13	14	16	18	20	23	25				

#### 8.7.2.4 Proceso de filtrado de bordes con bS igual a 4

Este proceso acepta como argumentos los valores de muestra  $p_i$  y  $q_i$  ( $i = 0..3$ ) de un solo conjunto de muestras situado en el borde que se desea filtrar, la variable chromaEdgeFlag, y los valores de las variables umbral  $\alpha$  y  $\beta$  para el conjunto de muestras que se pasan como argumentos, según se especifica en la subcláusula 8.7.2.

Este proceso genera como resultado los valores de las muestras filtradas  $p'_i$  y  $q'_i$  ( $i = 0..2$ ) del conjunto de valores de muestras.

Sean  $a_p$  y  $a_q$  las dos variables umbral definidas por las ecuaciones 8-339 y 8-340, respectivamente, en la subcláusula 8.7.2.3.

Las muestras filtradas  $p'_i$  ( $i = 0..2$ ) se calculan del modo siguiente.

– Si chromaEdgeFlag es igual a 0 y se cumple la siguiente condición,

$$a_p < \beta \ \&\& \ \text{Abs}(p_0 - q_0) < ((\alpha \gg 2) + 2) \quad (8-347)$$

las variables  $p'_0$ ,  $p'_1$ , y  $p'_2$  se calculan mediante la expresión

$$p'_0 = (p_2 + 2*p_1 + 2*p_0 + 2*q_0 + q_1 + 4) \gg 3 \quad (8-348)$$

$$p'_1 = (p_2 + p_1 + p_0 + q_0 + 2) \gg 2 \quad (8-349)$$

$$p'_2 = (2*p_3 + 3*p_2 + p_1 + p_0 + q_0 + 4) \gg 3 \quad (8-350)$$

– De lo contrario (chromaEdgeFlag es igual a 1 o no se cumple la condición expresada en la ecuación 8-347) las variables  $p'_0$ ,  $p'_1$ , y  $p'_2$  se calculan mediante las expresiones

$$p'_0 = (2*p_1 + p_0 + q_1 + 2) \gg 2 \quad (8-351)$$

$$p'_1 = p_1 \quad (8-352)$$



$$p'_2 = p_2 \quad (8-353)$$

Las muestras filtradas  $q'_i$  ( $i = 0..2$ ) se calculan del modo siguiente.

- Si `chromaEdgeFlag` es igual a 0 y se cumple la siguiente condición,

$$a_q < \beta \ \&\& \ \text{Abs}(p_0 - q_0) < ((\alpha \gg 2) + 2) \quad (8-354)$$

las variables  $q'_0$ ,  $q'_1$ , y  $q'_2$  se calculan mediante las expresiones

$$q'_0 = (p_1 + 2*p_0 + 2*q_0 + 2*q_1 + q_2 + 4) \gg 3 \quad (8-355)$$

$$q'_1 = (p_0 + q_0 + q_1 + q_2 + 2) \gg 2 \quad (8-356)$$

$$q'_2 = (2*q_3 + 3*q_2 + q_1 + q_0 + p_0 + 4) \gg 3 \quad (8-357)$$

- De lo contrario (`chromaEdgeFlag` es igual a 1 o no se cumple la condición expresada en la ecuación 8-354) las variables  $q'_0$ ,  $q'_1$ , y  $q'_2$  se calculan mediante las expresiones.

$$q'_0 = (2*q_1 + q_0 + p_1 + 2) \gg 2 \quad (8-358)$$

$$q'_1 = q_1 \quad (8-359)$$

$$q'_2 = q_2 \quad (8-360)$$

## 9 Proceso de análisis sintáctico

Este proceso acepta como argumentos los bits de la RBSP.

Este proceso genera como resultado valores de elementos sintácticos.

Se llama a este proceso cuando el descriptor de un elemento sintáctico en los cuadros de sintaxis que figuran en la subcláusula 7.3 es igual a  $ue(v)$ ,  $me(v)$ ,  $se(v)$ ,  $te(v)$  (véase subcláusula 9.1),  $ce(v)$  (véase subcláusula 9.2), o  $ae(v)$  (véase subcláusula 9.3).

### 9.1 Proceso de análisis sintáctico de códigos Exp-Golomb

Se llama a este proceso cuando el descriptor de un elemento sintáctico en los cuadros sintácticos de la subcláusula 7.3 es igual a  $ue(v)$ ,  $me(v)$ ,  $se(v)$ , o  $te(v)$ . Se llama a este proceso únicamente para elementos sintácticos de las subcláusulas 7.3.4 y 7.3.5 cuando `entropy_coding_mode_flag` es igual a 0.

Este proceso acepta como argumentos los bits de la RBSP.

Este proceso genera como resultado valores de elementos sintácticos.

Los elementos sintácticos codificados como  $ue(v)$ ,  $me(v)$ , o  $se(v)$  están codificados con Exp-Golomb. Los elementos sintácticos codificados como  $te(v)$  están codificados con Exp-Golomb truncado. El proceso de análisis sintáctico para esos elementos sintácticos empieza con la lectura de bits a partir de la posición actual del tren de bits hasta el primer bit distinto de 0, inclusive, y se cuenta el número de bits que están primero y que son iguales a 0, lo cual es equivalente a lo siguiente.

```

leadingZeroBits = -1;
for( b = 0; !b; leadingZeroBits++ )
    b = read_bits( 1 )

```

La variable `codeNum` se asigna mediante la siguiente expresión:

$$\text{codeNum} = 2^{\text{leadingZeroBits}} - 1 + \text{read\_bits}( \text{leadingZeroBits} )$$

donde el valor que devuelve `read_bits( leadingZeroBits )` se interpreta como una representación binaria de un entero sin signo de modo que el primer bit escrito es el más significativo.

El cuadro 9-1 ilustra la estructura del código Exp-Golomb, separando la cadena de bits "prefijo" y "sufijo". Los bits "prefijo" son los bits que se analizan en el seudocódigo anterior para calcular

leadingZeroBits, y aparecen como 0 ó 1 en la columna cadena de bits del cuadro 9-1. Los bits "sufijo" son aquellos bits que se analizan para calcular codeNum y aparecen como  $x_i$  en el cuadro 9-1, donde el valor de  $i$  está entre 0 y leadingZeroBits - 1, inclusive. Cada  $x_i$  puede valer 0 ó 1.

**Cuadro 9-1 – Cadena de bits con bits "prefijo" y "sufijo" y asignación de la gama de valores de codeNum (informativo)**

Forma de la cadena de bits	Valores posibles de codeNum
1	0
0 1 $x_0$	1-2
0 0 1 $x_1 x_0$	3-6
0 0 0 1 $x_2 x_1 x_0$	7-14
0 0 0 0 1 $x_3 x_2 x_1 x_0$	15-30
0 0 0 0 0 1 $x_4 x_3 x_2 x_1 x_0$	31-62
...	...

El cuadro 9-2 ilustra explícitamente la asignación de cadenas de bits a valores codeNum.

**Cuadro 9-2 – Cadenas de bits Exp-Golomb y codeNum en forma explícita y utilizadas como ue(v) (informativo)**

Cadena de bits	codeNum
1	0
0 1 0	1
0 1 1	2
0 0 1 0 0	3
0 0 1 0 1	4
0 0 1 1 0	5
0 0 1 1 1	6
0 0 0 1 0 0 0	7
0 0 0 1 0 0 1	8
0 0 0 1 0 1 0	9
...	...

El valor de elementos sintácticos se calcula en función del descriptor, del modo siguiente.

- Si el elemento sintáctico se codifica como ue(v), el valor del elemento sintáctico es igual a codeNum.
- De lo contrario, si el elemento sintáctico se codifica como se(v), para calcular el valor del elemento sintáctico se llama al proceso de mapeado de códigos Exp-Golomb con signo, descrito en la subcláusula 9.1.1, al que se le pasa como argumento codeNum.
- De lo contrario, si el elemento sintáctico se codifica como me(v), para calcular el valor del elemento sintáctico se llama al proceso de mapeado del patrón de bloques codificados descrito en la subcláusula 9.1.2, al que se le pasa como argumento codeNum.

- De lo contrario (el elemento sintáctico está codificado como  $te(v)$ ), se determina en primer lugar los valores posibles del elemento sintáctico. Éste toma valores entre 0 y  $x$ , siendo  $x$  mayor o igual que 1, y el cálculo de valor del elemento sintáctico depende del valor de  $x$ .
  - Si  $x$  es mayor que 1,  $codeNum$  y el valor del elemento sintáctico se calculan de la misma manera que los elementos sintácticos codificados como  $ue(v)$ .
  - De lo contrario ( $x$  es igual a 1), el proceso de análisis de  $codeNum$ , que es igual al valor del elemento sintáctico, es equivalente a la siguiente expresión:

```
b = read_bits( 1 )

codeNum = !b
```

### 9.1.1 Proceso de mapeado de códigos Exp-Golomb con signo

Este proceso acepta como argumento  $codeNum$ , según se describe en la subcláusula 9.1.

Este proceso genera como resultado el valor de un elemento sintáctico codificado como  $se(v)$ .

Se asigna a  $codeNum$  el valor del elemento sintáctico por orden creciente de su valor absoluto y de manera que el valor positivo se asigna antes que el negativo. En el cuadro 9-3 se muestra la fórmula de asignación.

**Cuadro 9-3 – Asignación a  $codeNum$  del elemento sintáctico para elementos sintácticos codificados mediante Exp-Golomb con signo  $se(v)$**

$codeNum$	valor del elemento sintáctico
0	0
1	1
2	-1
3	2
4	-2
5	3
6	-3
$k$	$(-1)^{k+1} \text{Ceil}(k \div 2)$

### 9.1.2 Proceso de mapeado del patrón de bloques codificados

Este proceso acepta como argumento  $codeNum$ , según se especifica en la subcláusula 9.1.

Este proceso genera como resultado el valor  $coded\_block\_pattern$  del elemento sintáctico codificado como  $me(v)$ .

En el cuadro 9-4 se muestra la asignación a  $codeNum$  del  $coded\_block\_pattern$  en función de si el modo de predicción de macrobloques es  $Intra\_4x4$  o  $Inter$ .

**Cuadro 9-4 – Asignación a codeNum de valores del coded\_block\_pattern para los modos de predicción de macrobloque**

codeNum	coded_block_pattern	
	Intra_4x4	Inter
0	47	0
1	31	16
2	15	1
3	0	2
4	23	4
5	27	8
6	29	32
7	30	3
8	7	5
9	11	10
10	13	12
11	14	15
12	39	47
13	43	7
14	45	11
15	46	13
16	16	14
17	3	6
18	5	9
19	10	31
20	12	35
21	19	37
22	21	42
23	26	44
24	28	33
25	35	34
26	37	36
27	42	40
28	44	39
29	1	43
30	2	45
31	4	46

**Cuadro 9-4 – Asignación a codeNum de valores del coded\_block\_pattern para los modos de predicción de macrobloque**

32	8	17
33	17	18
34	18	20
35	20	24
36	24	19
37	6	21
38	9	26
39	22	28
40	25	23
41	32	27
42	33	29
43	34	30
44	36	22
45	40	25
46	38	38
47	41	41

## 9.2 Proceso de análisis CAVLC de niveles de coeficientes de transformada

Se llama a este proceso para analizar elementos sintácticos cuyo descriptor es  $ce(v)$  en la subcláusula 7.3.5.3.1 y cuyo  $entropy\_coding\_mode\_flag$  es igual a 0.

Este proceso acepta como argumentos los bits de datos de sector, el número máximo de niveles de coeficiente de transformada distinto de cero  $maxNumCoeff$ , el índice del bloque  $luma4x4BlkIdx$  o el índice del bloque croma  $chroma4x4BlkIdx$  del bloque de niveles de coeficientes de transformada considerado.

Este proceso genera como resultado la lista  $coeffLevel$  que contiene los niveles de coeficientes de transformada del bloque  $luma$  con índice de bloque  $luma4x4BlkIdx$  o el bloque croma con índice de bloque  $chroma4x4BlkIdx$ .

Este proceso consta de los siguientes pasos:

1. Se ponen a cero todos los coeficientes de transformada con índices comprendidos entre 0 y  $maxNumCoeff - 1$  en la lista  $coeffLevel$ .
2. Se calcula el número total de niveles de coeficientes de transformada distintos de cero  $TotalCoeff(coeff\_token)$  y el número de niveles de coeficientes de transformada de valor uno finales  $TrailingOnes(coeff\_token)$ , para lo cual se analiza sintácticamente  $coeff\_token$  (véase subcláusula 9.2.1) del modo siguiente.

- Si el número de niveles de coeficientes de transformada distintos de cero  $TotalCoeff( coeff\_token )$  es igual a 0, se devuelve la lista  $coeffLevel$  que contiene los 0 y no se realiza ningún otro paso.
- De lo contrario, se llevan a cabo los siguientes pasos:
  - a. Se analizan  $trailing\_ones\_sign\_flag$ ,  $level\_prefix$ , y  $level\_suffix$  para calcular los niveles de coeficientes de transformada distintos de cero (véase subcláusula 9.2.2).
  - b. Se analizan  $total\_zeros$  y  $run\_before$  para calcular las series de niveles de coeficiente de transformada cero antes de cada nivel de coeficiente de transformada distinto de cero (véase subcláusula 9.2.3).
  - c. Se combina en la lista  $coeffLevel$  la información sobre niveles y series (véase subcláusula 9.2.4).

### 9.2.1 Proceso de análisis del número total de niveles de coeficientes de transformada y de valor uno finales

Este proceso acepta como argumento los bits de datos de sector, el número máximo de niveles de coeficientes de transformada distintos de cero  $maxNumCoeff$ , el índice del bloque  $luma4x4BlkIdx$  o el índice del bloque  $chroma4x4BlkIdx$  del bloque de transformada considerado.

Este proceso genera como resultado  $TotalCoeff( coeff\_token )$  y  $TrailingOnes( coeff\_token )$ .

Se decodifica el elemento sintáctico  $coeff\_token$  con uno de los cinco VLC especificados en las últimas cinco columnas de la derecha del cuadro 9-5. Cada VLC especifica  $TotalCoeff( coeff\_token )$  y  $TrailingOnes( coeff\_token )$  correspondiente a una determinada palabra de código  $coeff\_token$ . La selección del VLC depende de la variable  $nC$  que se calcula del modo siguiente.

- Si el proceso de análisis de CAVLC se aplica a un  $ChromaDCLevel$ ,  $nC$  se pone a  $-1$ .
- De lo contrario se aplica lo siguiente.
  - Cuando el proceso de análisis CAVLC se aplica a un  $Intra16x16DCLevel$ ,  $luma4x4BlkIdx$  se pone a 0.
  - Se calculan las variables  $blkA$  y  $blkB$ .
    - Si el proceso de análisis CAVLC se aplica a  $Intra16x16DCLevel$ ,  $Intra16x16ACLevel$ , o  $LumaLevel$ , se llama al proceso descrito en la subcláusula 6.4.7.3, pasándole como argumento  $luma4x4BlkIdx$  y el resultado se asigna a  $mbAddrA$ ,  $mbAddrB$ ,  $luma4x4BlkIdxA$ , y  $luma4x4BlkIdxB$ . Se asigna a  $blkA$  el bloque  $luma\ 4x4$  definido por  $mbAddrA\luma4x4BlkIdxA$  y se asigna a  $blkB$  el bloque  $luma\ 4x4$  definido por  $mbAddrB\luma4x4BlkIdxB$ .
    - De lo contrario (el proceso de análisis CAVLC se aplica a un  $ChromaACLevel$ ), se llama al proceso descrito en la subcláusula 6.4.7.4, pasándole como argumento  $chroma4x4BlkIdx$ , y el resultado se asigna a  $mbAddrA$ ,  $mbAddrB$ ,  $chroma4x4BlkIdxA$  y  $chroma4x4BlkIdxB$ . Se asigna a  $blkA$  el bloque croma  $4x4$  definido por  $mbAddrA\iCbCr\chroma4x4BlkIdxA$  y se asigna a  $blkB$  el bloque croma  $4x4$  definido por  $mbAddrB\iCbCr\chroma4x4BlkIdxB$ .
  - Sean  $nA$  y  $nB$ , respectivamente, el número de niveles de coeficientes de transformada distintos de cero (dado por  $TotalCoeff( coeff\_token )$ ) en el bloque de niveles de coeficiente de transformada  $blkA$  situado a la izquierda del bloque considerado y el bloque de niveles de coeficientes de transformada  $blkB$  situado encima del bloque considerado.
  - Siendo  $N$  igual a  $A$  y  $B$  en  $mbAddrN$ ,  $blkN$  y  $nN$ , se aplica lo siguiente.
    - Si se cumple alguna de las siguientes condiciones,  $nN$  se pone a 0.

- mbAddrN no está disponible.
- El macrobloque considerado está codificado en modo de predicción intra, constrained\_intra\_pred\_flag es igual a 1 y mbAddrN está codificado en modo de predicción inter y se utilizan particiones de datos de sectores (el valor de nal\_unit\_type está entre 2 y 4, inclusive).
- El macrobloque mbAddrN tiene mb\_type igual a P\_Skip o B\_Skip.
- Todos los niveles de coeficientes de transformada residual c.a. del bloque adyacente blkN son iguales a 0 debido a que el correspondiente bit de CodedBlockPatternLuma o CodedBlockPatternChroma es igual a 0
- De lo contrario, si mbAddrN es un macrobloque I\_PCM, nN se pone a 16.
- De lo contrario, nN se pone al valor TotalCoeff( coeff\_token ) del bloque adyacente blkN.

NOTA – Los valores nA y nB que se obtienen mediante TotalCoeff( coeff\_token ) no incluyen los niveles de coeficientes de transformada c.c. en macrobloques Intra\_16x16 o los niveles de coeficientes de transformada c.c. en bloques cromas, dado que estos niveles de coeficiente de transformada se decodifican por separado. Cuando el bloque situado por encima o la izquierda forma parte de un macrobloque Intra\_16x16, o es un bloque cromas, nA y nB son iguales al número de niveles de coeficientes de transformada c.a. distintos de cero decodificados.

NOTA – Cuando se analiza un Intra16x16DCLevel, los valores nA y nB se basan en el número de niveles de coeficientes de transformada distintos de cero en bloques 4x4 adyacentes y no el número de niveles de coeficientes de transformada c.c. distintos de cero en bloques 16x16 adyacentes.

- Se calcula la variable nC a partir de los valores de nA y nB.
  - Si se dispone de mbAddrA y mbAddrB, el valor de la variable nC es  $(nA + nB + 1) \gg 1$ .
  - De lo contrario (mbAddrA no está disponible o mbAddrB no está disponible), el valor de la variable nC es  $nA + nB$ .

El valor de TotalCoeff( coeff\_token ) que resulta de decodificar coeff\_token estará entre 0 y maxNumCoeff, inclusive.

**Cuadro 9-5 – Mapeado de coeff\_token en TotalCoeff( coeff\_token ) y TrailingOnes( coeff\_token )**

TrailingOnes ( coeff_token )	TotalCoeff ( coeff_token )	0 <= nC < 2	2 <= nC < 4	4 <= nC < 8	8 <= nC	nC == -1
0	0	1	11	1111	0000 11	01
0	1	0001 01	0010 11	0011 11	0000 00	0001 11
1	1	01	10	1110	0000 01	1
0	2	0000 0111	0001 11	0010 11	0001 00	0001 00
1	2	0001 00	0011 1	0111 1	0001 01	0001 10
2	2	001	011	1101	0001 10	001
0	3	0000 0011 1	0000 111	0010 00	0010 00	0000 11
1	3	0000 0110	0010 10	0110 0	0010 01	0000 011
2	3	0000 101	0010 01	0111 0	0010 10	0000 010
3	3	0001 1	0101	1100	0010 11	0001 01
0	4	0000 0001 11	0000 0111	0001 111	0011 00	0000 10
1	4	0000 0011 0	0001 10	0101 0	0011 01	0000 0011
2	4	0000 0101	0001 01	0101 1	0011 10	0000 0010
3	4	0000 11	0100	1011	0011 11	0000 000
0	5	0000 0000 111	0000 0100	0001 011	0100 00	–
1	5	0000 0001 10	0000 110	0100 0	0100 01	–
2	5	0000 0010 1	0000 101	0100 1	0100 10	–
3	5	0000 100	0011 0	1010	0100 11	–
0	6	0000 0000 0111 1	0000 0011 1	0001 001	0101 00	–
1	6	0000 0000 110	0000 0110	0011 10	0101 01	–
2	6	0000 0001 01	0000 0101	0011 01	0101 10	–
3	6	0000 0100	0010 00	1001	0101 11	–
0	7	0000 0000 0101 1	0000 0001 111	0001 000	0110 00	–
1	7	0000 0000 0111 0	0000 0011 0	0010 10	0110 01	–
2	7	0000 0000 101	0000 0010 1	0010 01	0110 10	–
3	7	0000 0010 0	0001 00	1000	0110 11	–
0	8	0000 0000 0100 0	0000 0001 011	0000 1111	0111 00	–
1	8	0000 0000 0101 0	0000 0001 110	0001 110	0111 01	–
2	8	0000 0000 0110 1	0000 0001 101	0001 101	0111 10	–
3	8	0000 0001 00	0000 100	0110 1	0111 11	–
0	9	0000 0000 0011 11	0000 0000 1111	0000 1011	1000 00	–
1	9	0000 0000 0011 10	0000 0001 010	0000 1110	1000 01	–
2	9	0000 0000 0100 1	0000 0001 001	0001 010	1000 10	–
3	9	0000 0000 100	0000 0010 0	0011 00	1000 11	–
0	10	0000 0000 0010 11	0000 0000 1011	0000 0111 1	1001 00	–
1	10	0000 0000 0010 10	0000 0000 1110	0000 1010	1001 01	–
2	10	0000 0000 0011 01	0000 0000 1101	0000 1101	1001 10	–
3	10	0000 0000 0110 0	0000 0001 100	0001 100	1001 11	–
0	11	0000 0000 0001 111	0000 0000 1000	0000 0101 1	1010 00	–
1	11	0000 0000 0001 110	0000 0000 1010	0000 0111 0	1010 01	–
2	11	0000 0000 0010 01	0000 0000 1001	0000 1001	1010 10	–
3	11	0000 0000 0011 00	0000 0001 000	0000 1100	1010 11	–
0	12	0000 0000 0001 011	0000 0000 0111 1	0000 0100 0	1011 00	–



**Cuadro 9-5 – Mapeado de coeff\_token en TotalCoeff( coeff\_token ) y TrailingOnes( coeff\_token )**

1	12	0000 0000 0001 010	0000 0000 0111 0	0000 0101 0	1011 01	–
2	12	0000 0000 0001 101	0000 0000 0110 1	0000 0110 1	1011 10	–
3	12	0000 0000 0010 00	0000 0000 1100	0000 1000	1011 11	–
0	13	0000 0000 0000 1111	0000 0000 0101 1	0000 0011 01	1100 00	–
1	13	0000 0000 0000 001	0000 0000 0101 0	0000 0011 1	1100 01	–
2	13	0000 0000 0001 001	0000 0000 0100 1	0000 0100 1	1100 10	–
3	13	0000 0000 0001 100	0000 0000 0110 0	0000 0110 0	1100 11	–
0	14	0000 0000 0000 1011	0000 0000 0011 1	0000 0010 01	1101 00	–
1	14	0000 0000 0000 1110	0000 0000 0010 11	0000 0011 00	1101 01	–
2	14	0000 0000 0000 1101	0000 0000 0011 0	0000 0010 11	1101 10	–
3	14	0000 0000 0001 000	0000 0000 0100 0	0000 0010 10	1101 11	–
0	15	0000 0000 0000 0111	0000 0000 0010 01	0000 0001 01	1110 00	–
1	15	0000 0000 0000 1010	0000 0000 0010 00	0000 0010 00	1110 01	–
2	15	0000 0000 0000 1001	0000 0000 0010 10	0000 0001 11	1110 10	–
3	15	0000 0000 0000 1100	0000 0000 0000 1	0000 0001 10	1110 11	–
0	16	0000 0000 0000 0100	0000 0000 0001 11	0000 0000 01	1111 00	–
1	16	0000 0000 0000 0110	0000 0000 0001 10	0000 0001 00	1111 01	–
2	16	0000 0000 0000 0101	0000 0000 0001 01	0000 0000 11	1111 10	–
3	16	0000 0000 0000 1000	0000 0000 0001 00	0000 0000 10	1111 11	–

### 9.2.2 Proceso de análisis de información sobre niveles

Este proceso acepta como argumentos los bits de los datos de sector, el número de niveles de coeficiente de transformada distintos de cero TotalCoeff( coeff\_token ), y el número de niveles de coeficiente de transformada de valor uno finales TrailingOnes( coeff\_token ).

Este proceso genera como resultado una lista con nivel de nombres que contiene los niveles de coeficiente de transformada.

En primer lugar el índice *i* se pone a cero. A continuación, para decodificar los niveles de coeficiente de transformada con bits uno de cola (si los hubiere), se aplica reiteradamente el siguiente procedimiento un número de veces igual a TrailingOnes( coeff\_token ):

- Se decodifica y calcula el elemento sintáctico de 1 bit trailing\_ones\_sign\_flag según se describe a continuación.
  - Si trailing\_ones\_sign\_flag es igual a 0, se asigna el valor +1 al nivel[ *i* ].
  - De lo contrario (trailing\_ones\_sign\_flag es igual a 1), se asigna el valor –1 al nivel[ *i* ].
- Se incrementa en 1 el índice *i*.

Después de decodificar los niveles de coeficiente de transformada de valor uno finales, se inicializa la variable suffixLength.

- Si TotalCoeff( coeff\_token ) es mayor que 10 y TrailingOnes( coeff\_token ) es menor que 3, suffixLength se pone a 1.
- De lo contrario (TotalCoeff( coeff\_token ) es menor o igual que 10 o TrailingOnes( coeff\_token ) es igual a 3), suffixLength se pone a 0.

Para decodificar los niveles restantes (si los hubiere), se aplica reiteradamente el siguiente procedimiento un número de veces igual a  $(\text{TotalCoeff}(\text{coeff\_token}) - \text{TrailingOnes}(\text{coeff\_token}))$ :

- Se decodifica el elemento sintáctico `level_prefix` mediante el VLC especificado en el cuadro 9-6.
- Se asigna a la variable `levelSuffixSize` el valor de la variable `suffixLength`, salvo en los siguientes dos casos.
  - Cuando `level_prefix` es igual a 14 y `suffixLength` es igual a 0, `levelSuffixSize` se pone a 4.
  - Cuando `level_prefix` es igual a 15, `levelSuffixSize` se pone a 12.
- Se decodifica el elemento sintáctico `level_suffix` del modo siguiente.
  - Si `levelSuffixSize` es mayor que 0, el elemento sintáctico `level_suffix` se decodifica como una representación entera sin signo  $u(v)$  del `levelSuffixSize` bits.
  - De lo contrario (`levelSuffixSize` es igual a 0), el elemento sintáctico `level_suffix` se supone que es igual a 0.
- Se asigna a la variable `levelCode` el valor  $(\text{level\_prefix} \ll \text{suffixLength}) + \text{level\_suffix}$ .
- Cuando `level_prefix` es igual a 15 y `suffixLength` es igual a 0, `levelCode` se incrementa en 15.
- Cuando el índice  $i$  es igual a  $\text{TrailingOnes}(\text{coeff\_token})$  y  $\text{TrailingOnes}(\text{coeff\_token})$  es menor que 3, `levelCode` se incrementa en 2.
- Se calcula la variable `level[ i ]` del modo siguiente.
  - Si `levelCode` es un número par, se asigna a `level[ i ]` el valor  $(\text{levelCode} + 2) \gg 1$ .
  - De lo contrario, se asigna a `level[ i ]` el valor  $(-\text{levelCode} - 1) \gg 1$ .
- Cuando `suffixLength` es igual a 0, `suffixLength` se pone a 1.
- Cuando el valor absoluto de `level[ i ]` es mayor que  $(3 \ll (\text{suffixLength} - 1))$  y `suffixLength` es menor que 6, `suffixLength` se incrementa en 1.
- El índice  $i$  se incrementa en 1.

**Cuadro 9-6 – Palabras de código para level\_prefix**

level_prefix	Cadena de bits
0	1
1	01
2	001
3	0001
4	0000 1
5	0000 01
6	0000 001
7	0000 0001
8	0000 0000 1
9	0000 0000 01
10	0000 0000 001
11	0000 0000 0001
12	0000 0000 0000 1
13	0000 0000 0000 01
14	0000 0000 0000 001
15	0000 0000 0000 0001

### 9.2.3 Proceso de análisis de información sobre series

Este proceso acepta como argumentos los bits de datos de sector, el número de coeficientes de transformada distintos de cero TotalCoeff( coeff\_token ) y el número máximo de niveles de coeficientes de transformada distintos de cero maxNumCoeff.

Este proceso genera como resultado una lista de series de niveles de coeficiente de transformada de valor cero que preceden a niveles de coeficientes de transformada distintos de cero, lista que se denomina serie.

En primer lugar, el índice  $i$  se pone a 0.

Se calcula la variable zerosLeft del modo siguiente:

- Si el número de niveles de coeficientes de transformada distintos de cero TotalCoeff( coeff\_token ) es igual al número máximo de niveles de coeficiente de transformada distintos de cero maxNumCoeff, la variable zerosLeft se pone a 0.
- De lo contrario (el número de niveles de coeficientes de transformada distintos de cero TotalCoeff( coeff\_token ) es menor que el número máximo de niveles de coeficiente de transformada distintos de cero maxNumCoeff), se decodifica total\_zeros y zerosLeft se pone al valor de éste.

Se obtiene el VLC con el que se decodificará total\_zeros:

- Si maxNumCoeff es igual a 4, se utiliza uno de los VLC especificados en el cuadro 9-9.
- De lo contrario (maxNumCoeff es distinto de 4), se utilizan los VLC de los cuadros 9-7 y 9-8.

Se aplica reiteradamente el siguiente procedimiento un número de veces igual a ( TotalCoeff( coeff\_token ) – 1 ):

- Se calcula la variable run[  $i$  ].

- Si zerosLeft es mayor que cero, el valor de run\_before se decodifica a partir del cuadro 9-10 y del valor de zerosLeft. Se asigna a run[ i ] el valor de run\_before.
- De lo contrario (zerosLeft es igual a 0), run[ i ] se pone a 0.
- Se resta a zerosLeft el valor de run[ i ] y el resultado se asigna a zerosLeft. El resultado de la resta será mayor o igual que 0.
- Se incrementa el índice i en 1.

Por último se asigna a run[ i ] el valor de zerosLeft.

**Cuadro 9-7 – total\_zeros correspondiente a bloques 4x4 para valores de TotalCoeff( coeff\_token ) entre 1 y 7**

total_zeros	TotalCoeff( coeff_token )						
	1	2	3	4	5	6	7
0	1	111	0101	0001 1	0101	0000 01	0000 01
1	011	110	111	111	0100	0000 1	0000 1
2	010	101	110	0101	0011	111	101
3	0011	100	101	0100	111	110	100
4	0010	011	0100	110	110	101	011
5	0001 1	0101	0011	101	101	100	11
6	0001 0	0100	100	100	100	011	010
7	0000 11	0011	011	0011	011	010	0001
8	0000 10	0010	0010	011	0010	0001	001
9	0000 011	0001 1	0001 1	0010	0000 1	001	0000 00
10	0000 010	0001 0	0001 0	0001 0	0001	0000 00	
11	0000 0011	0000 11	0000 01	0000 1	0000 0		
12	0000 0010	0000 10	0000 1	0000 0			
13	0000 0001 1	0000 01	0000 00				
14	0000 0001 0	0000 00					
15	0000 0000 1						

**Cuadro 9-8 – total\_zeros correspondiente a bloques 4x4 para valores de TotalCoeff( coeff\_token ) entre 8 y 15**

total_zeros	TotalCoeff( coeff_token )							
	8	9	10	11	12	13	14	15
0	0000 01	0000 01	0000 1	0000	0000	000	00	0
1	0001	0000 00	0000 0	0001	0001	001	01	1
2	0000 1	0001	001	001	01	1	1	
3	011	11	11	010	1	01		
4	11	10	10	1	001			
5	10	001	01	011				
6	010	01	0001					
7	001	0000 1						
8	0000 00							

**Cuadro 9-9 – total\_zeros correspondiente a bloques 2x2 DC croma**

total_zeros	TotalCoeff( coeff_token )		
	1	2	3
0	1	1	1
1	01	01	0
2	001	00	
3	000		

**Cuadro 9-10 – run\_before en función de zerosLeft**

run_before	zerosLeft						
	1	2	3	4	5	6	>6
0	1	1	11	11	11	11	111
1	0	01	10	10	10	000	110
2	–	00	01	01	011	001	101
3	–	–	00	001	010	011	100
4	–	–	–	000	001	010	011
5	–	–	–	–	000	101	010
6	–	–	–	–	–	100	001
7	–	–	–	–	–	–	0001
8		–	–	–	–	–	00001
9	–	–	–	–	–	–	000001
10	–	–	–	–	–	–	0000001
11	–	–	–	–	–	–	00000001
12	–	–	–	–	–	–	000000001
13	–	–	–	–	–	–	0000000001
14	–	–	–	–	–	–	00000000001

#### 9.2.4 Combinación de la información sobre niveles y series

Este proceso acepta como argumentos una lista de niveles de coeficientes de transformada denominada nivel, una lista de series denominada serie y el número de niveles de coeficientes de transformada distintos de cero TotalCoeff( coeff\_token ).

Este proceso genera como resultado una lista de niveles de coeficiente de transformada, coeffLevel.

La variable coeffNum se pone a –1 y el índice i se pone a ( TotalCoeff( coeff\_token ) – 1 ). Se aplica TotalCoeff( coeff\_token ) veces el siguiente procedimiento:

- coeffNum se incrementa en run[ i ] + 1.
- Se asigna a coeffLevel[ coeffNum ] el valor de level[ i ].
- El índice i se disminuye en 1.

#### 9.3 Proceso de análisis CABAC de datos de sector

Se llama este proceso al analizar elementos sintácticos cuyo descriptor es ae(v) en las subcláusulas 7.3.4 y 7.3.5 y entropy\_coding\_mode\_flag es igual a 1.

Este proceso acepta como argumento una petición del valor de un elemento sintáctico y los valores de los elementos sintácticos analizados antes.

Este proceso genera como resultado el valor del elemento sintáctico.

Al iniciar el análisis de los datos de sector de un sector descrito en la subcláusula 7.3.4, se llama al proceso de inicialización del proceso de análisis CABAC que se describe en la subcláusula 9.3.1.

El análisis de los elementos sintácticos consiste en lo siguiente:

Para cada valor solicitado de un elemento sintáctico se calcula su binarización, según se describe en la subcláusula 9.3.2.

La binarización del elemento sintáctico y la secuencia de bins analizados determina el flujo del proceso de decodificación, como se describe en la subcláusula 9.3.3.

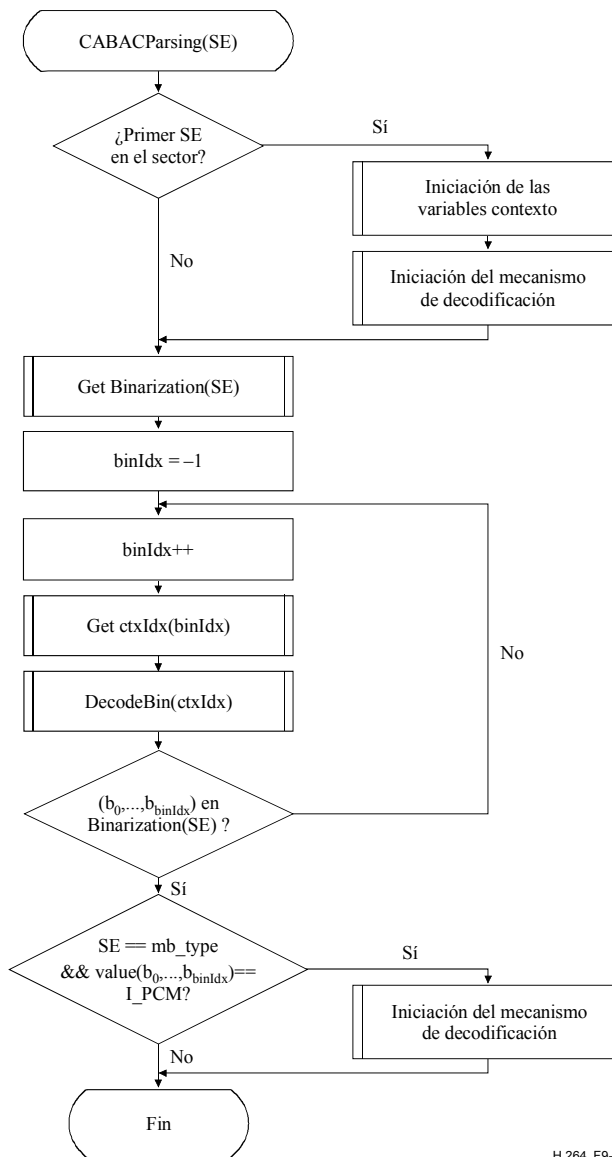
Para cada bin de la binarización del elemento sintáctico, que está indexado mediante la variable `binIdx`, se calcula el índice de contexto `ctxIdx`, como se describe en la subcláusula 9.3.3.1.

Para cada `ctxIdx` se llama al proceso de decodificación aritmética que se describe en la subcláusula 9.3.3.2.

La secuencia resultante ( $b_0 \dots b_{\text{binIdx}}$ ) de bins analizados se compara con el conjunto de cadenas bin obtenidas mediante el proceso de binarización después de codificar cada bin. Cuando la secuencia corresponde con una cadena bin del conjunto determinado, se asigna al elemento sintáctico el correspondiente valor.

En caso de que la petición del valor de elemento sintáctico sea del elemento sintáctico `mb_type` y el valor decodificado de `mb_type` sea `I_PCM`, se inicializará el mecanismo de decodificación después de decodificar el `pcm_alignment_zero_bit` y todos los datos `pcm_byte`, como se describe en la subcláusula 9.3.1.2.

El diagrama de flujo de la figura 9-1 ilustra todo el proceso de análisis CABAC, donde la abreviatura SE significa elemento sintáctico.



**Figura 9-1 – Ilustración del proceso de análisis CABAC de un elemento sintáctico SE (informativo)**

### 9.3.1 Proceso de inicialización

Este proceso genera como resultado las variables internas CABAC inicializadas.

Se llama a los procesos descritos en las subcláusulas 9.3.1.1 y 9.3.1.2 al iniciar el análisis de los datos de sector de un sector descrito en la subcláusula 7.3.4.

Asimismo, se llama al proceso descrito en la subcláusula 9.3.1.2 después de decodificar el `pcm_alignment_zero_bit` y todos los datos `pcm_byte` correspondientes a un macrobloque del tipo `I_PCM`.

#### 9.3.1.1 Proceso de inicialización de variables contexto

Este proceso genera como resultado variables contexto CABAC inicializadas, y cuyo índice es `ctxIdx`.

Los cuadros 9-12 a 9-23 contienen los valores de las variables `n` y `m` utilizadas en la inicialización de variables contexto que se asignan a todos los elementos sintácticos en las subcláusulas 7.3.4 y 7.3.5, excepto al indicador fin de sector.

Para cada variable contexto se inicializan dos variables, pStateIdx y valMPS.

NOTA – La variable pStateIdx corresponde al índice de estado de probabilidad y la variable valMPS corresponde al valor del símbolo más probable, según se describe en la subcláusula 9.3.3.2.

Los dos valores asignados a pStateIdx y valMPS para la inicialización se calculan a partir de SliceQP<sub>Y</sub>, y ésta se calcula mediante la ecuación 7-16. Dados los dos elementos del cuadro (m, n),

1. preCtxState = Clip3( 1, 126, ( ( m \* SliceQP<sub>Y</sub> ) >> 4 ) + n )
2. if( preCtxState <= 63 ) {  
    pStateIdx = 63 - preCtxState  
    valMPS = 0  
} else {  
    pStateIdx = preCtxState - 64  
    valMPS = 1  
}

En el cuadro 9-11 se enumeran el ctxIdx para el cual se necesita inicialización para cada tipo de sector. Asimismo se indica el número de cuadro que incluye los valores de m y n necesarios para la inicialización. Para los sectores de tipos P, SP y B la inicialización también depende del valor del elemento sintáctico cabac\_init\_idc. Obsérvese que los nombres de los elementos sintácticos no afectan al proceso de inicialización.



**Cuadro 9-11 – Asociación de ctxIdx y elementos sintácticos para cada tipo de sector en el proceso de inicialización**

	Elemento sintáctico	Cuadro	Tipo de sector			
			SI	I	P, SP	B
slice_data()	mb_skip_flag	Cuadro 9-13 Cuadro 9-14			11-13	24-26
	mb_field_decoding_flag	Cuadro 9-18	70-72	70-72	70-72	70-72
	mb_type	Cuadro 9-12 Cuadro 9-13 Cuadro 9-14	0-10	3-10	14-20	27-35
macroblock_layer()	coded_block_pattern (luma)	Cuadro 9-18	73-76	73-76	73-76	73-76
	coded_block_pattern (chroma)	Cuadro 9-18	77-84	77-84	77-84	77-84
	mb_qp_delta	Cuadro 9-17	60-63	60-63	60-63	60-63
	prev_intra4x4_pred_mode_flag	Cuadro 9-17	68	68	68	68
mb_pred()	rem_intra4x4_pred_mode	Cuadro 9-17	69	69	69	69
	intra_chroma_pred_mode	Cuadro 9-17	64-67	64-67	64-67	64-67
	ref_idx_10	Cuadro 9-16			54-59	54-59
mb_pred() and sub_mb_pred()	ref_idx_11	Cuadro 9-16				54-59
	mvd_10[ ][ ][ 0 ]	Cuadro 9-15			40-46	40-46
	mvd_11[ ][ ][ 0 ]	Cuadro 9-15				40-46
	mvd_10[ ][ ][ 1 ]	Cuadro 9-15			47-53	47-53
	mvd_11[ ][ ][ 1 ]	Cuadro 9-15				47-53
	sub_mb_type	Cuadro 9-13 Cuadro 9-14			21-23	36-39
residual_block_cabac()	coded_block_flag	Cuadro 9-18	85-104	85-104	85-104	85-104
	significant_coeff_flag[ ]	Cuadro 9-19 Cuadro 9-22	105-165, 277-337	105-165, 277-337	105-165, 277-337	105-165, 277-337
	last_significant_coeff_flag[ ]	Cuadro 9-20 Cuadro 9-23	166-226, 338-398	166-226, 338-398	166-226, 338-398	166-226, 338-398
	coeff_abs_level_minus1[ ]	Cuadro 9-21	227-275	227-275	227-275	227-275

NOTA – El valor 276 de ctxIdx corresponde a end\_of\_slice\_flag y al bin de mb\_type, que indica un macrobloque de tipo I\_PCM. El proceso de decodificación descrito en la subcláusula 9.3.3.2.4 se aplica a ctxIdx igual a 276. Sin embargo, este proceso de decodificación también se puede implementar mediante el proceso de decodificación descrito en la subcláusula 9.3.3.2.1. En ese caso, los valores iniciales asociados a ctxIdx igual a 276 se especifican mediante pStateIdx = 63 y valMPS = 0, donde pStateIdx = 63 representa un estado de probabilidad no adaptativa.

**Cuadro 9-12 – Valores de las variables m y n para ctxIdx entre 0 y 10**

Variables de inicialización	ctxIdx										
	0	1	2	3	4	5	6	7	8	9	10
<b>m</b>	20	2	3	20	2	3	-28	-23	-6	-1	7
<b>n</b>	-15	54	74	-15	54	74	127	104	53	54	51

**Cuadro 9-13 – Valores de las variables m y n para ctxIdx entre 11 y 23**

Valor de cabac_init_idc	Variables de inicialización	ctxIdx												
		11	12	13	14	15	16	17	18	19	20	21	22	23
0	m	23	23	21	1	0	-37	5	-13	-11	1	12	-4	17
	n	33	2	0	9	49	118	57	78	65	62	49	73	50
1	m	22	34	16	-2	4	-29	2	-6	-13	5	9	-3	10
	n	25	0	0	9	41	118	65	71	79	52	50	70	54
2	m	29	25	14	-10	-3	-27	26	-4	-24	5	6	-17	14
	n	16	0	0	51	62	99	16	85	102	57	57	73	57

**Cuadro 9-14 – Valores de las variables m y n para ctxIdx entre 24 y 39**

Valor de cabac_init_idc	Variables de inicialización	ctxIdx															
		24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39
0	m	18	9	29	26	16	9	-46	-20	1	-13	-11	1	-6	-17	-6	9
	n	64	43	0	67	90	104	127	104	67	78	65	62	86	95	61	45
1	m	26	19	40	57	41	26	-45	-15	-4	-6	-13	5	6	-13	0	8
	n	34	22	0	2	36	69	127	101	76	71	79	52	69	90	52	43
2	m	20	20	29	54	37	12	-32	-22	-2	-4	-24	5	-6	-14	-6	4
	n	40	10	0	0	42	97	127	117	74	85	102	57	93	88	44	55

**Cuadro 9-15 – Valores de las variables m y n para ctxIdx entre 40 y 53**

Valor de cabac_init_idc	Variables de inicialización	ctxIdx													
		40	41	42	43	44	45	46	47	48	49	50	51	52	53
0	m	-3	-6	-11	6	7	-5	2	0	-3	-10	5	4	-3	0
	n	69	81	96	55	67	86	88	58	76	94	54	69	81	88
1	m	-2	-5	-10	2	2	-3	-3	1	-3	-6	0	-3	-7	-5
	n	69	82	96	59	75	87	100	56	74	85	59	81	86	95
2	m	-11	-15	-21	19	20	4	6	1	-5	-13	5	6	-3	-1
	n	89	103	116	57	58	84	96	63	85	106	63	75	90	101

**Cuadro 9-16 – Valores de las variables m y n para ctxIdx entre 54 y 59**

Valor de cabac_init_idc	Variables de inicialización	ctxIdx					
		54	55	56	57	58	59
0	m	-7	-5	-4	-5	-7	1
	n	67	74	74	80	72	58
1	m	-1	-1	1	-2	-5	0
	n	66	77	70	86	72	61
2	m	3	-4	-2	-12	-7	1
	n	55	79	75	97	50	60

**Cuadro 9-17 – Valores de las variables m y n para ctxIdx entre 60 y 69**

Variables de inicialización	ctxIdx									
	60	61	62	63	64	65	66	67	68	69
<b>m</b>	0	0	0	0	-9	4	0	-7	13	3
<b>n</b>	41	63	63	63	83	86	97	72	41	62

**Cuadro 9-18 – Valores de las variables m y n para ctxIdx entre 70 y 104**

ctxIdx	Sectoros I y SI		Valor de cabac_init_idc						ctxIdx	Sectoros I y SI		Valores de cabac_init_idc					
			0		1		2					0		1		2	
	m	n	m	n	m	n	m	n		m	n	m	n	m	n		
<b>70</b>	0	11	0	45	13	15	7	34	<b>88</b>	-11	115	-13	108	-4	92	5	78
<b>71</b>	1	55	-4	78	7	51	-9	88	<b>89</b>	-12	63	-3	46	0	39	-6	55
<b>72</b>	0	69	-3	96	2	80	-20	127	<b>90</b>	-2	68	-1	65	0	65	4	61
<b>73</b>	-17	127	-27	126	-39	127	-36	127	<b>91</b>	-15	84	-1	57	-15	84	-14	83
<b>74</b>	-13	102	-28	98	-18	91	-17	91	<b>92</b>	-13	104	-9	93	-35	127	-37	127
<b>75</b>	0	82	-25	101	-17	96	-14	95	<b>93</b>	-3	70	-3	74	-2	73	-5	79
<b>76</b>	-7	74	-23	67	-26	81	-25	84	<b>94</b>	-8	93	-9	92	-12	104	-11	104
<b>77</b>	-21	107	-28	82	-35	98	-25	86	<b>95</b>	-10	90	-8	87	-9	91	-11	91
<b>78</b>	-27	127	-20	94	-24	102	-12	89	<b>96</b>	-30	127	-23	126	-31	127	-30	127
<b>79</b>	-31	127	-16	83	-23	97	-17	91	<b>97</b>	-1	74	5	54	3	55	0	65
<b>80</b>	-24	127	-22	110	-27	119	-31	127	<b>98</b>	-6	97	6	60	7	56	-2	79
<b>81</b>	-18	95	-21	91	-24	99	-14	76	<b>99</b>	-7	91	6	59	7	55	0	72
<b>82</b>	-27	127	-18	102	-21	110	-18	103	<b>100</b>	-20	127	6	69	8	61	-4	92
<b>83</b>	-21	114	-13	93	-18	102	-13	90	<b>101</b>	-4	56	-1	48	-3	53	-6	56
<b>84</b>	-30	127	-29	127	-36	127	-37	127	<b>102</b>	-5	82	0	68	0	68	3	68
<b>85</b>	-17	123	-7	92	0	80	11	80	<b>103</b>	-7	76	-4	69	-7	74	-8	71
<b>86</b>	-12	115	-5	89	-5	89	5	76	<b>104</b>	-22	125	-8	88	-9	88	-13	98
<b>87</b>	-16	122	-7	96	-7	94	2	84									

**Cuadro 9-19 – Valores de las variables m y n para ctxIdx entre 105 y 165**

ctxIdx	Sector I y SI		Valor de cabac_init_idc						ctxIdx	Sector I y SI		Valores de cabac_init_idc					
			0		1		2					0		1		2	
	m	n	m	n	m	n	m	n		m	n	m	n	m	n		
105	-7	93	-2	85	-13	103	-4	86	136	-13	101	5	53	0	58	-5	75
106	-11	87	-6	78	-13	91	-12	88	137	-13	91	-2	61	-1	60	-8	80
107	-3	77	-1	75	-9	89	-5	82	138	-12	94	0	56	-3	61	-21	83
108	-5	71	-7	77	-14	92	-3	72	139	-10	88	0	56	-8	67	-21	64
109	-4	63	2	54	-8	76	-4	67	140	-16	84	-13	63	-25	84	-13	31
110	-4	68	5	50	-12	87	-8	72	141	-10	86	-5	60	-14	74	-25	64
111	-12	84	-3	68	-23	110	-16	89	142	-7	83	-1	62	-5	65	-29	94
112	-7	62	1	50	-24	105	-9	69	143	-13	87	4	57	5	52	9	75
113	-7	65	6	42	-10	78	-1	59	144	-19	94	-6	69	2	57	17	63
114	8	61	-4	81	-20	112	5	66	145	1	70	4	57	0	61	-8	74
115	5	56	1	63	-17	99	4	57	146	0	72	14	39	-9	69	-5	35
116	-2	66	-4	70	-78	127	-4	71	147	-5	74	4	51	-11	70	-2	27
117	1	64	0	67	-70	127	-2	71	148	18	59	13	68	18	55	13	91
118	0	61	2	57	-50	127	2	58	149	-8	102	3	64	-4	71	3	65
119	-2	78	-2	76	-46	127	-1	74	150	-15	100	1	61	0	58	-7	69
120	1	50	11	35	-4	66	-4	44	151	0	95	9	63	7	61	8	77
121	7	52	4	64	-5	78	-1	69	152	-4	75	7	50	9	41	-10	66
122	10	35	1	61	-4	71	0	62	153	2	72	16	39	18	25	3	62
123	0	44	11	35	-8	72	-7	51	154	-11	75	5	44	9	32	-3	68
124	11	38	18	25	2	59	-4	47	155	-3	71	4	52	5	43	-20	81
125	1	45	12	24	-1	55	-6	42	156	15	46	11	48	9	47	0	30
126	0	46	13	29	-7	70	-3	41	157	-13	69	-5	60	0	44	1	7
127	5	44	13	36	-6	75	-6	53	158	0	62	-1	59	0	51	-3	23
128	31	17	-10	93	-8	89	8	76	159	0	65	0	59	2	46	-21	74
129	1	51	-7	73	-34	119	-9	78	160	21	37	22	33	19	38	16	66
130	7	50	-2	73	-3	75	-11	83	161	-15	72	5	44	-4	66	-23	124
131	28	19	13	46	32	20	9	52	162	9	57	14	43	15	38	17	37
132	16	33	9	49	30	22	0	67	163	16	54	-1	78	12	42	44	-18
133	14	62	-7	100	-44	127	-5	90	164	0	62	0	60	9	34	50	-34
134	-13	108	9	53	0	54	1	67	165	12	72	9	69	0	89	-22	127
135	-15	100	2	53	-5	61	-15	72									

**Cuadro 9-20 – Valores de las variables m y n para ctxIdx entre 166 y 226**

ctxIdx	Sectoros I y SI		Valor de cabac_init_idc						ctxIdx	Sectoros I y SI		Valores de cabac_init_idc					
			0		1		2					0		1		2	
	m	n	m	n	m	n	m	n		m	n	m	n	m	n		
<b>166</b>	24	0	11	28	4	45	4	39	<b>197</b>	26	-17	28	3	36	-28	28	-3
<b>167</b>	15	9	2	40	10	28	0	42	<b>198</b>	30	-25	28	4	38	-28	24	10
<b>168</b>	8	25	3	44	10	31	7	34	<b>199</b>	28	-20	32	0	38	-27	27	0
<b>169</b>	13	18	0	49	33	-11	11	29	<b>200</b>	33	-23	34	-1	34	-18	34	-14
<b>170</b>	15	9	0	46	52	-43	8	31	<b>201</b>	37	-27	30	6	35	-16	52	-44
<b>171</b>	13	19	2	44	18	15	6	37	<b>202</b>	33	-23	30	6	34	-14	39	-24
<b>172</b>	10	37	2	51	28	0	7	42	<b>203</b>	40	-28	32	9	32	-8	19	17
<b>173</b>	12	18	0	47	35	-22	3	40	<b>204</b>	38	-17	31	19	37	-6	31	25
<b>174</b>	6	29	4	39	38	-25	8	33	<b>205</b>	33	-11	26	27	35	0	36	29
<b>175</b>	20	33	2	62	34	0	13	43	<b>206</b>	40	-15	26	30	30	10	24	33
<b>176</b>	15	30	6	46	39	-18	13	36	<b>207</b>	41	-6	37	20	28	18	34	15
<b>177</b>	4	45	0	54	32	-12	4	47	<b>208</b>	38	1	28	34	26	25	30	20
<b>178</b>	1	58	3	54	102	-94	3	55	<b>209</b>	41	17	17	70	29	41	22	73
<b>179</b>	0	62	2	58	0	0	2	58	<b>210</b>	30	-6	1	67	0	75	20	34
<b>180</b>	7	61	4	63	56	-15	6	60	<b>211</b>	27	3	5	59	2	72	19	31
<b>181</b>	12	38	6	51	33	-4	8	44	<b>212</b>	26	22	9	67	8	77	27	44
<b>182</b>	11	45	6	57	29	10	11	44	<b>213</b>	37	-16	16	30	14	35	19	16
<b>183</b>	15	39	7	53	37	-5	14	42	<b>214</b>	35	-4	18	32	18	31	15	36
<b>184</b>	11	42	6	52	51	-29	7	48	<b>215</b>	38	-8	18	35	17	35	15	36
<b>185</b>	13	44	6	55	39	-9	4	56	<b>216</b>	38	-3	22	29	21	30	21	28
<b>186</b>	16	45	11	45	52	-34	4	52	<b>217</b>	37	3	24	31	17	45	25	21
<b>187</b>	12	41	14	36	69	-58	13	37	<b>218</b>	38	5	23	38	20	42	30	20
<b>188</b>	10	49	8	53	67	-63	9	49	<b>219</b>	42	0	18	43	18	45	31	12
<b>189</b>	30	34	-1	82	44	-5	19	58	<b>220</b>	35	16	20	41	27	26	27	16
<b>190</b>	18	42	7	55	32	7	10	48	<b>221</b>	39	22	11	63	16	54	24	42
<b>191</b>	10	55	-3	78	55	-29	12	45	<b>222</b>	14	48	9	59	7	66	0	93
<b>192</b>	17	51	15	46	32	1	0	69	<b>223</b>	27	37	9	64	16	56	14	56
<b>193</b>	17	46	22	31	0	0	20	33	<b>224</b>	21	60	-1	94	11	73	15	57
<b>194</b>	0	89	-1	84	27	36	8	63	<b>225</b>	12	68	-2	89	10	67	26	38
<b>195</b>	26	-19	25	7	33	-25	35	-18	<b>226</b>	2	97	-9	108	-10	116	-24	127
<b>196</b>	22	-17	30	-7	34	-30	33	-25									

**Cuadro 9-21 – Valores de las variables m y n para ctxIdx entre 227 y 275**

ctxIdx	Sector I y SI		Valor de cabac_init_idc						ctxIdx	Sector I y SI		Valores de cabac_init_idc					
			0		1		2					0		1		2	
	m	n	m	n	m	n	m	n		m	n	m	n	m	n		
227	-3	71	-6	76	-23	112	-24	115	252	-12	73	-6	55	-16	72	-14	75
228	-6	42	-2	44	-15	71	-22	82	253	-8	76	0	58	-7	69	-10	79
229	-5	50	0	45	-7	61	-9	62	254	-7	80	0	64	-4	69	-9	83
230	-3	54	0	52	0	53	0	53	255	-9	88	-3	74	-5	74	-12	92
231	-2	62	-3	64	-5	66	0	59	256	-17	110	-10	90	-9	86	-18	108
232	0	58	-2	59	-11	77	-14	85	257	-11	97	0	70	2	66	-4	79
233	1	63	-4	70	-9	80	-13	89	258	-20	84	-4	29	-9	34	-22	69
234	-2	72	-4	75	-9	84	-13	94	259	-11	79	5	31	1	32	-16	75
235	-1	74	-8	82	-10	87	-11	92	260	-6	73	7	42	11	31	-2	58
236	-9	91	-17	102	-34	127	-29	127	261	-4	74	1	59	5	52	1	58
237	-5	67	-9	77	-21	101	-21	100	262	-13	86	-2	58	-2	55	-13	78
238	-5	27	3	24	-3	39	-14	57	263	-13	96	-3	72	-2	67	-9	83
239	-3	39	0	42	-5	53	-12	67	264	-11	97	-3	81	0	73	-4	81
240	-2	44	0	48	-7	61	-11	71	265	-19	117	-11	97	-8	89	-13	99
241	0	46	0	55	-11	75	-10	77	266	-8	78	0	58	3	52	-13	81
242	-16	64	-6	59	-15	77	-21	85	267	-5	33	8	5	7	4	-6	38
243	-8	68	-7	71	-17	91	-16	88	268	-4	48	10	14	10	8	-13	62
244	-10	78	-12	83	-25	107	-23	104	269	-2	53	14	18	17	8	-6	58
245	-6	77	-11	87	-25	111	-15	98	270	-3	62	13	27	16	19	-2	59
246	-10	86	-30	119	-28	122	-37	127	271	-13	71	2	40	3	37	-16	73
247	-12	92	1	58	-11	76	-10	82	272	-10	79	0	58	-1	61	-10	76
248	-15	55	-3	29	-10	44	-8	48	273	-12	86	-3	70	-5	73	-13	86
249	-10	60	-1	36	-10	52	-8	61	274	-13	90	-6	79	-1	70	-9	83
250	-6	62	1	38	-10	57	-8	66	275	-14	97	-8	85	-4	78	-10	87
251	-4	65	2	43	-9	58	-7	70									

**Cuadro 9-22 – Valores de las variables m y n para ctxIdx entre 277 y 337**

ctxIdx	Sector I y SI		Valor de cabac_init_idc						ctxIdx	Sector I y SI		Valores de cabac_init_idc					
			0		1		2					0		1		2	
	m	n	m	n	m	n	m	n		m	n	m	n	m	n		
277	-6	93	-13	106	-21	126	-22	127	308	-16	96	-1	51	-16	77	-10	67
278	-6	84	-16	106	-23	124	-25	127	309	-7	88	7	49	-2	64	1	68
279	-8	79	-10	87	-20	110	-25	120	310	-8	85	8	52	2	61	0	77
280	0	66	-21	114	-26	126	-27	127	311	-7	85	9	41	-6	67	2	64
281	-1	71	-18	110	-25	124	-19	114	312	-9	85	6	47	-3	64	0	68
282	0	62	-14	98	-17	105	-23	117	313	-13	88	2	55	2	57	-5	78
283	-2	60	-22	110	-27	121	-25	118	314	4	66	13	41	-3	65	7	55
284	-2	59	-21	106	-27	117	-26	117	315	-3	77	10	44	-3	66	5	59
285	-5	75	-18	103	-17	102	-24	113	316	-3	76	6	50	0	62	2	65
286	-3	62	-21	107	-26	117	-28	118	317	-6	76	5	53	9	51	14	54
287	-4	58	-23	108	-27	116	-31	120	318	10	58	13	49	-1	66	15	44
288	-9	66	-26	112	-33	122	-37	124	319	-1	76	4	63	-2	71	5	60
289	-1	79	-10	96	-10	95	-10	94	320	-1	83	6	64	-2	75	2	70
290	0	71	-12	95	-14	100	-15	102	321	-7	99	-2	69	-1	70	-2	76
291	3	68	-5	91	-8	95	-10	99	322	-14	95	-2	59	-9	72	-18	86
292	10	44	-9	93	-17	111	-13	106	323	2	95	6	70	14	60	12	70
293	-7	62	-22	94	-28	114	-50	127	324	0	76	10	44	16	37	5	64
294	15	36	-5	86	-6	89	-5	92	325	-5	74	9	31	0	47	-12	70
295	14	40	9	67	-2	80	17	57	326	0	70	12	43	18	35	11	55
296	16	27	-4	80	-4	82	-5	86	327	-11	75	3	53	11	37	5	56
297	12	29	-10	85	-9	85	-13	94	328	1	68	14	34	12	41	0	69
298	1	44	-1	70	-8	81	-12	91	329	0	65	10	38	10	41	2	65
299	20	36	7	60	-1	72	-2	77	330	-14	73	-3	52	2	48	-6	74
300	18	32	9	58	5	64	0	71	331	3	62	13	40	12	41	5	54
301	5	42	5	61	1	67	-1	73	332	4	62	17	32	13	41	7	54
302	1	48	12	50	9	56	4	64	333	-1	68	7	44	0	59	-6	76
303	10	62	15	50	0	69	-7	81	334	-13	75	7	38	3	50	-11	82
304	17	46	18	49	1	69	5	64	335	11	55	13	50	19	40	-2	77
305	9	64	17	54	7	69	15	57	336	5	64	10	57	3	66	-2	77
306	-12	104	10	41	-7	69	1	67	337	12	70	26	43	18	50	25	42
307	-11	97	7	46	-6	67	0	68									

**Cuadro 9-23 – Valores de las variables m y n para ctxIdx entre 338 y 398**

ctxIdx	Sectores I y SI		Valor de cabac_init_idc						ctxIdx	Sectores I y SI		Valores de cabac_init_idc					
			0		1		2					0		1		2	
	m	n	m	n	m	n	m	n		m	n	m	n	m	n		
338	15	6	14	11	19	-6	17	-13	369	32	-26	31	-4	40	-37	37	-17
339	6	19	11	14	18	-6	16	-9	370	37	-30	27	6	38	-30	32	1
340	7	16	9	11	14	0	17	-12	371	44	-32	34	8	46	-33	34	15
341	12	14	18	11	26	-12	27	-21	372	34	-18	30	10	42	-30	29	15
342	18	13	21	9	31	-16	37	-30	373	34	-15	24	22	40	-24	24	25
343	13	11	23	-2	33	-25	41	-40	374	40	-15	33	19	49	-29	34	22
344	13	15	32	-15	33	-22	42	-41	375	33	-7	22	32	38	-12	31	16
345	15	16	32	-15	37	-28	48	-47	376	35	-5	26	31	40	-10	35	18
346	12	23	34	-21	39	-30	39	-32	377	33	0	21	41	38	-3	31	28
347	13	23	39	-23	42	-30	46	-40	378	38	2	26	44	46	-5	33	41
348	15	20	42	-33	47	-42	52	-51	379	33	13	23	47	31	20	36	28
349	14	26	41	-31	45	-36	46	-41	380	23	35	16	65	29	30	27	47
350	14	44	46	-28	49	-34	52	-39	381	13	58	14	71	25	44	21	62
351	17	40	38	-12	41	-17	43	-19	382	29	-3	8	60	12	48	18	31
352	17	47	21	29	32	9	32	11	383	26	0	6	63	11	49	19	26
353	24	17	45	-24	69	-71	61	-55	384	22	30	17	65	26	45	36	24
354	21	21	53	-45	63	-63	56	-46	385	31	-7	21	24	22	22	24	23
355	25	22	48	-26	66	-64	62	-50	386	35	-15	23	20	23	22	27	16
356	31	27	65	-43	77	-74	81	-67	387	34	-3	26	23	27	21	24	30
357	22	29	43	-19	54	-39	45	-20	388	34	3	27	32	33	20	31	29
358	19	35	39	-10	52	-35	35	-2	389	36	-1	28	23	26	28	22	41
359	14	50	30	9	41	-10	28	15	390	34	5	28	24	30	24	22	42
360	10	57	18	26	36	0	34	1	391	32	11	23	40	27	34	16	60
361	7	63	20	27	40	-1	39	1	392	35	5	24	32	18	42	15	52
362	-2	77	0	57	30	14	30	17	393	34	12	28	29	25	39	14	60
363	-4	82	-14	82	28	26	20	38	394	39	11	23	42	18	50	3	78
364	-3	94	-5	75	23	37	18	45	395	30	29	19	57	12	70	-16	123
365	9	69	-19	97	12	55	15	54	396	34	26	22	53	21	54	21	53
366	-12	109	-35	125	11	65	0	79	397	29	39	22	61	14	71	22	56
367	36	-35	27	0	37	-33	36	-16	398	19	66	11	86	11	83	25	61
368	36	-34	28	0	39	-36	37	-14									

### 9.3.1.2 Proceso de inicialización del mecanismo de decodificación aritmética

Se llama a este proceso antes de decodificar el primer macrobloque de un sector o después de decodificar el pcm\_alignment\_zero\_bit y todos los datos pcm\_byte de un macrobloque de tipo I\_PCM.

El resultado de aplicar este proceso es la inicialización de los registros del mecanismo de decodificación codIRange y codIOffset; los dos registros tienen una precisión de 16 bits.

El estado del mecanismo de decodificación aritmética se representa mediante las variables codIRange y codIOffset. En el procedimiento de inicialización del proceso de decodificación aritmética, codIRange se pone a 0x01FE y se asigna a codIOffset el valor que devuelve read\_bits(9), que se interpreta como una representación binaria de 9 bits de un entero sin signo, siendo el primer bit escrito el más significativo.



NOTA – La descripción del mecanismo de decodificación aritmética en esta Recomendación | Norma Internacional utiliza una precisión de registro de 16 bits. Sin embargo, la precisión de registro mínima para las variables `codIRange` y `codIOffset` es 9 bits.

### 9.3.2 Proceso de binarización

Este proceso acepta como argumento una petición de un elemento sintáctico.

Este proceso genera como resultado la binarización del elemento sintáctico, `maxBinIdxCtx`, `ctxIdxOffset` y `bypassFlag`.

En el cuadro 9-24 se especifica el tipo de proceso de binarización y `maxBinIdxCtx`, y `ctxIdxOffset` correspondientes a cada elemento sintáctico.

En las subcláusulas 9.3.2.1 a 9.3.2.4 se especifican, respectivamente, el proceso de binarización unario (U), el proceso de binarización unario truncado (TU), el proceso de binarización unario/Exp-Golomb de orden  $k$ -ésimo concatenados (UEGk), y el proceso de binarización de longitud fija (FL). En las subcláusulas 9.3.2.5 a 9.3.2.7 se especifican otras binarizaciones.

Excepto para los sectores I, las binarizaciones del elemento sintáctico `mb_type`, según se especifica en la subcláusula 9.3.2.5, consta de cadenas bin formadas por la concatenación de cadenas de bit prefijo y sufijo. La binarización UEGk que se describe en 9.3.2.3, utilizada para la binarización de elementos sintácticos `mvd_IX` ( $X = 0, 1$ ) y `coeff_abs_level_minus1`, y la binarización de `coded_block_pattern`, también consisten en la concatenación de cadenas de bits prefijo y sufijo. Para estos procesos de binarización las cadenas de bits prefijo y sufijo se indexan por separado mediante la variable `binIdx`, según se especifica más adelante en la subcláusula 9.3.3. Los dos conjuntos de cadenas de bits prefijo y sufijo se denominan parte prefijo de binarización y parte sufijo de binarización.

Cada binarización o parte binarización de un elemento sintáctico tiene asignado un valor específico de la variable traslación de índice de contexto (`ctxIdxOffset`) y un valor específico de la variable `maxBinIdxCtx`, como se indica en el cuadro 9-24. Cuando en el cuadro 9-24 se especifican dos valores para cada una de estas variables de un mismo elemento sintáctico, el valor de la fila superior corresponde a la parte prefijo y el valor de la fila inferior corresponde a la parte sufijo de la binarización del correspondiente elemento sintáctico.

La utilización del proceso `DecodeBypass` y la variable `bypassFlag` se calculan del modo siguiente.

- Si `ctxIdxOffset` no tiene ningún valor asignado para la correspondiente binarización o parte de binarización en el cuadro 9-24 indicado como "na", todos los bins de la cadena de bits de la correspondiente binarización o de la parte prefijo/sufijo de binarización se decodificarán llamando al proceso `DecodeBypass`, descrito en la subcláusula 9.3.3.2.3. En ese caso, `bypassFlag` se pone a 1; así pues, mediante `bypassFlag` se indica que para analizar el valor del bin de la cadena de bits se aplica el proceso `DecodeBypass`.
- De lo contrario, para cada valor posible de `binIdx` hasta el valor especificado de `MaxBinIdxCtx` en el cuadro 9-24, el valor específico de la variable `ctxIdx` se especifica en la subcláusula 9.3.3. `bypassFlag` se pone a 0.

El índice de contexto `ctxIdx` toma valores entre 0 y 398, inclusive. El valor asignado a `ctxIdxOffset` especifica el valor más pequeño, de los posibles valores de `ctxIdx`, asignado a la correspondiente binarización o parte binarización del elemento sintáctico.

`ctxIdx = ctxIdxOffset = 276` se asigna al elemento sintáctico `end_of_slice_flag` y al bin de `mb_type`, el cual especifica el tipo de macrobloque `I_PCM` como se describe más adelante en la subcláusula 9.3.3.1. Para analizar sintácticamente el valor del correspondiente bin del tren de bits, se aplica el proceso de decodificación aritmética de decisiones antes de la terminación (`DecodeTerminate`) que se describe en la subcláusula 9.3.3.2.4.

NOTA – Los bins de mb\_type en sectores I y los bins del sufijo del mb\_type en sectores SI que corresponden con el mismo valor de binIdx tienen el mismo ctxIdx. El último bin del prefijo de mb\_type y el primer bin del sufijo de mb\_type en sectores P, SP y B pueden tener el mismo ctxIdx.

**Cuadro 9-24 – Elementos sintácticos y tipos de binarización asociados, maxBinIdxCtx y ctxIdxOffset**

Elemento sintáctico	Tipo de binarización	maxBinIdxCtx	ctxIdxOffset
mb_type (sólo sectores SI)	Prefijo y sufijo el indicado en la subcláusula 9.3.2.5	Prefijo: 0 Sufijo: 6	Prefijo: 0 Sufijo: 3
mb_type (sólo sectores I)	El indicado en la subcláusula 9.3.2.5	6	3
mb_skip_flag (sólo sectores P, SP)	FL, cMax=1	0	11
mb_type (sólo sectores P, SP)	Prefijo y sufijo el indicado en la subcláusula 9.3.2.5	Prefijo: 2 Sufijo: 5	Prefijo: 14 Sufijo: 17
sub_mb_type (sólo sectores P, SP)	El indicado en la subcláusula 9.3.2.5	2	21
mb_skip_flag (sólo sectores B)	FL, cMax=1	0	24
mb_type (sólo sectores B)	Prefijo y sufijo el indicado en la subcláusula 9.3.2.5	Prefijo: 3 Sufijo: 5	Prefijo: 27 Sufijo: 32
sub_mb_type (sólo sectores B)	Prefijo y sufijo el indicado en la subcláusula 9.3.2.5	3	36
mvd_10[ ][ ][ 0 ], mvd_11[ ][ ][ 0 ]	Prefijo y sufijo indicados por UEG3 con signedValFlag=1, uCoff=9	Prefijo: 4 Sufijo: na	Prefijo: 40 Sufijo: na (se utiliza DecodeBypass)
mvd_10[ ][ ][ 1 ], mvd_11[ ][ ][ 1 ]		Prefijo: 4 Sufijo: na	Prefijo: 47 Sufijo: na (se utiliza DecodeBypass)
ref_idx_10, ref_idx_11	U	2	54
mb_qp_delta	El indicado en la subcláusula 9.3.2.7	2	60
intra_chroma_pred_mode	TU, cMax=3	1	64
prev_intra4x4_pred_mode_flag	FL, cMax=1	0	68
rem_intra4x4_pred_mode	FL, cMax=7	0	69
mb_field_decoding_flag	FL, cMax=1	0	70
coded_block_pattern	Prefijo y sufijo el indicado en la subcláusula 9.3.2.6	Prefijo: 3 Sufijo: 1	Prefijo: 73 Sufijo: 77
coded_block_flag	FL, cMax=1	0	85
significant_coeff_flag (sólo bloques cuadro codificados)	FL, cMax=1	0	105

**Cuadro 9-24 – Elementos sintácticos y tipos de binarización asociados, maxBinIdxCtx y ctxIdxOffset**

Elemento sintáctico	Tipo de binarización	maxBinIdxCtx	ctxIdxOffset
last_significant_coeff_flag (sólo bloques cuadro codificados)	FL, cMax=1	0	166
coeff_abs_level_minus1	Prefijo y sufijo indicados por UEG0 con signedValFlag=0, uCoff=14	Prefijo: 1 Sufijo: na	Prefijo: 227 Sufijo: na, (se utiliza DecodeBypass)
coeff_sign_flag	FL, cMax=1	0	na, (se utiliza DecodeBypass)
end_of_slice_flag	FL, cMax=1	0	276
significant_coeff_flag (sólo bloques campo codificados)	FL, cMax=1	0	277
last_significant_coeff_flag (sólo bloques campo codificados)	FL, cMax=1	0	338

### 9.3.2.1 Proceso de binarización unario (U)

Este proceso acepta como argumento una petición de binarización U de un elemento sintáctico.

Este proceso genera como resultado la binarización U del elemento sintáctico.

La cadena bin de un elemento sintáctico que tiene el valor (entero sin signo) synElVal es una cadena de bits de longitud synElVal + 1 indexada mediante BinIdx. Los bins de binIdx menores que synElVal son iguales a 1. El bin con binIdx igual a synElVal es igual a 0.

El cuadro 9-25 ilustra las cadenas bin de la binarización de un elemento sintáctico.

**Cuadro 9-25 – Cadena bin de la binarización unaria (informativo)**

Valor del elemento sintáctico	Cadena bin					
0	0					
1	1	0				
2	1	1	0			
3	1	1	1	0		
4	1	1	1	1	0	
5	1	1	1	1	1	0
...						
binIdx	0	1	2	3	4	5

### 9.3.2.2 Proceso de binarización unaria truncada (TU)

Este proceso acepta como argumentos una petición de binarización TU de un elemento sintáctico y cMax.

Este proceso genera como resultado la binarización TU del elemento sintáctico.

Para valores (entero sin signo) del elemento sintáctico menores que  $cMax$ , se llama al proceso de binarización U descrito en la subcláusula 9.3.2.1. Cuando el valor del elemento sintáctico es igual a  $cMax$  la cadena bin es una cadena de bits de longitud  $cMax$  con todos los bins iguales a 1.

NOTA – Siempre se llama al proceso de binarización TU con el valor de  $cMax$  igual al valor posible más grande del elemento sintáctico que se está decodificando.

### 9.3.2.3 Proceso de binarización unaria/Exp-Golomb de orden k-ésimo concatenado (UEGk)

Este proceso acepta como argumentos una petición de binarización UEGk de un elemento sintáctico, `signedValFlag`, y `uCoff`.

Este proceso genera como resultado la binarización UEGk del elemento sintáctico.

Una cadena bin UEGk es una concatenación de una cadena de bits prefijo y una cadena de bits sufijo. Para calcular el prefijo de la binarización se llama al proceso de binarización TU para la parte prefijo  $\text{Min}( uCoff, \text{Abs}( \text{synElVal} ) )$  de un valor de elemento sintáctico `synElVal`, como se describe en la subcláusula 9.3.2.2, con  $cMax = uCoff$ , siendo  $uCoff > 0$ .

La cadena bin UEGk se calcula del modo siguiente:

- Si se cumple alguna de las condiciones siguientes, la cadena bin del elemento sintáctico con valor `synElVal` consta sólo de una cadena de bits prefijo:
  - `signedValFlag` es igual a 0 y la cadena de bits prefijo es distinta a la cadena de bits de longitud `uCoff` con todos los bits igual a 1;
  - `signedValFlag` es igual a 1 y la cadena de bits prefijo es igual a la cadena de bits que está formada por un solo bit con valor igual a 0.
- De lo contrario, la cadena bin de la parte sufijo UEGk de un valor elemento sintáctico `synElVal` se especifica mediante un proceso equivalente al siguiente pseudocódigo:

```
if( Abs( synElVal ) >= uCoff ) {
  sufS = Abs( synElVal ) - uCoff
  stopLoop = 0
  do {
    if( sufS >= ( 1 << k ) ) {
      put( 1 )
      sufS = sufS - ( 1 << k )
      k++
    } else {
      put( 0 )
      while( k-- )
        put( ( sufS >> k ) & 0x01 )
      stopLoop = 1
    }
  } while( !stopLoop )
}
if( signedValFlag && synElVal != 0 )
  if( synElVal > 0 )
    put( 0 )
  else
    put( 1 )
```

NOTA – El código Exp-Golomb de orden k-ésimo (EGk) utiliza los niveles lógicos 0 y 1 con una lógica inversa para parte unaria del código Exp-Golomb de orden 0-ésimo, según se especifica en la subcláusula 9.1.

### 9.3.2.4 Proceso de binarización de longitud fija (FL)

Este proceso acepta como argumentos una petición de binarización FL de un elemento sintáctico y  $cMax$ .

Este proceso genera como resultado la binarización FL del elemento sintáctico.

La binarización FL se realiza mediante una cadena bin de enteros sin signo de longitud `fixedLength` bits del valor del elemento sintáctico, siendo `fixedLength = Ceil( Log2( cMax + 1 ) )`. La indexación de bins de la binarización FL es tal que el `binIdx = 0` corresponde al bit menos significativo y el valor de `binIdx` aumenta a medida que el bit es más significativo.

### **9.3.2.5 Proceso de binarización de tipo macrobloque y tipo submacrobloque**

Este proceso acepta como argumento una petición de una binarización de elementos sintácticos `mb_type` o `sub_mb_type`.

Este proceso genera como resultado la binarización del elemento sintáctico.

En el cuadro 9-26 se especifica el esquema de binarización para codificar el tipo macrobloque en sectores I.

Para tipos de macrobloques en sectores SI, la binarización consiste en cadenas bin formadas a partir de una concatenación de una cadena de bits prefijo y sufijo, como se describe a continuación.

La cadena de bits prefijo consta de un solo bit, definido por  $b_0 = ((mb\_type == SI) ? 0 : 1)$ . Para el valor del elemento sintáctico cuyo  $b_0$  es igual a 0, la cadena bin sólo consta de una cadena de bits prefijo. Para el valor del elemento sintáctico cuyo  $b_0$  es igual a 1, la binarización viene dada por la concatenación del prefijo  $b_0$  y la cadena de bits sufijo especificada en el cuadro 9-26 para el tipo macrobloque en sectores SI, y cuyo índice es el valor del `mb_type` en sectores SI menos 1.

**Cuadro 9-26 – Binarización para tipo de macrobloque en sectores SI**

Valor (nombre) de mb_type	Cadena bin						
0 (I_4x4)	0						
1 (I_16x16_0_0_0)	1	0	0	0	0	0	
2 (I_16x16_1_0_0)	1	0	0	0	0	1	
3 (I_16x16_2_0_0)	1	0	0	0	1	0	
4 (I_16x16_3_0_0)	1	0	0	0	1	1	
5 (I_16x16_0_1_0)	1	0	0	1	0	0	0
6 (I_16x16_1_1_0)	1	0	0	1	0	0	1
7 (I_16x16_2_1_0)	1	0	0	1	0	1	0
8 (I_16x16_3_1_0)	1	0	0	1	0	1	1
9 (I_16x16_0_2_0)	1	0	0	1	1	0	0
10 (I_16x16_1_2_0)	1	0	0	1	1	0	1
11 (I_16x16_2_2_0)	1	0	0	1	1	1	0
12 (I_16x16_3_2_0)	1	0	0	1	1	1	1
13 (I_16x16_0_0_1)	1	0	1	0	0	0	
14 (I_16x16_1_0_1)	1	0	1	0	0	1	
15 (I_16x16_2_0_1)	1	0	1	0	1	0	
16 (I_16x16_3_0_1)	1	0	1	0	1	1	
17 (I_16x16_0_1_1)	1	0	1	1	0	0	0
18 (I_16x16_1_1_1)	1	0	1	1	0	0	1
19 (I_16x16_2_1_1)	1	0	1	1	0	1	0
20 (I_16x16_3_1_1)	1	0	1	1	0	1	1
21 (I_16x16_0_2_1)	1	0	1	1	1	0	0
22 (I_16x16_1_2_1)	1	0	1	1	1	0	1
23 (I_16x16_2_2_1)	1	0	1	1	1	1	0
24 (I_16x16_3_2_1)	1	0	1	1	1	1	1
25 (I_PCM)	1	1					
binIdx	0	1	2	3	4	5	6

En el cuadro 9-27 se especifica el esquema de binarización para tipos de macrobloques P en sectores P y SP y para macrobloques B en sectores B.

La cadena bin de tipos de macrobloque I en sectores P y SP para valores de mb\_type comprendidos entre 5 y 30 consta de la concatenación de un prefijo, que consiste en un solo bit cuyo valor es igual a 1 como se describe en el cuadro 9-27, y un sufijo como se describe en el cuadro 9-26, y cuyo índice es el valor de mb\_type menos 5.

El valor de mb\_type igual a 4 (P\_8x8ref0) está prohibido.

Para macrobloques de tipo I en sectores B (valores de mb\_type entre 23 y 48) la binarización consta de cadenas bin creadas por la concatenación de una cadena de bits prefijo, como se especifica en el cuadro 9-27, y cadenas de bit sufijo, como se especifica en el cuadro 9-26, y el índice se obtiene de restar 23 al valor de mb\_type.

**Cuadro 9-27 – Binarización de tipo de macrobloque en sectores P, SP y B**

Tipo de sector	Valor (nombre) de mb_type	Cadena bin						
Sector P, SP	0 (P_L0_16x16)	0	0	0				
	1 (P_L0_L0_16x8)	0	1	1				
	2 (P_L0_L0_8x16)	0	1	0				
	3 (P_8x8)	0	0	1				
	4 (P_8x8ref0)	na						
	5 a 30 (Intra, solo prefijo)	1						
Sector B	0 (B_Direct_16x16)	0						
	1 (B_L0_16x16)	1	0	0				
	2 (B_L1_16x16)	1	0	1				
	3 (B_Bi_16x16)	1	1	0	0	0	0	
	4 (B_L0_L0_16x8)	1	1	0	0	0	1	
	5 (B_L0_L0_8x16)	1	1	0	0	1	0	
	6 (B_L1_L1_16x8)	1	1	0	0	1	1	
	7 (B_L1_L1_8x16)	1	1	0	1	0	0	
	8 (B_L0_L1_16x8)	1	1	0	1	0	1	
	9 (B_L0_L1_8x16)	1	1	0	1	1	0	
	10 (B_L1_L0_16x8)	1	1	0	1	1	1	
	11 (B_L1_L0_8x16)	1	1	1	1	1	0	
	12 (B_L0_Bi_16x8)	1	1	1	0	0	0	0
	13 (B_L0_Bi_8x16)	1	1	1	0	0	0	1
	14 (B_L1_Bi_16x8)	1	1	1	0	0	1	0
	15 (B_L1_Bi_8x16)	1	1	1	0	0	1	1
	16 (B_Bi_L0_16x8)	1	1	1	0	1	0	0
	17 (B_Bi_L0_8x16)	1	1	1	0	1	0	1
	18 (B_Bi_L1_16x8)	1	1	1	0	1	1	0
	19 (B_Bi_L1_8x16)	1	1	1	0	1	1	1
	20 (B_Bi_Bi_16x8)	1	1	1	1	0	0	0
	21 (B_Bi_Bi_8x16)	1	1	1	1	0	0	1
	22 (B_8x8)	1	1	1	1	1	1	
23 to 48 (Intra, solo prefijo)	1	1	1	1	0	1		
binIdx	0	1	2	3	4	5	6	

En el cuadro 9-28 se especifica la binarización de sub\_mb\_type para sectores P, SP y B.

**Cuadro 9-28 – Binarización para tipos de submacrobloques en sectores P, SP y B**

Tipo de sector	Valor (nombre) de mb_type	Cadena de bins					
Sector P, SP	0 (P_L0_8x8)	1					
	1 (P_L0_8x4)	0	0				
	2 (P_L0_4x8)	0	1	1			
	3 (P_L0_4x4)	0	1	0			
Sector B	0 (B_Direct_8x8)	0					
	1 (B_L0_8x8)	1	0	0			
	2 (B_L1_8x8)	1	0	1			
	3 (B_Bi_8x8)	1	1	0	0	0	
	4 (B_L0_8x4)	1	1	0	0	1	
	5 (B_L0_4x8)	1	1	0	1	0	
	6 (B_L1_8x4)	1	1	0	1	1	
	7 (B_L1_4x8)	1	1	1	0	0	0
	8 (B_Bi_8x4)	1	1	1	0	0	1
	9 (B_Bi_4x8)	1	1	1	0	1	0
	10 (B_L0_4x4)	1	1	1	0	1	1
	11 (B_L1_4x4)	1	1	1	1	0	
12 (B_Bi_4x4)	1	1	1	1	1		
binIdx		0	1	2	3	4	5

### 9.3.2.6 Proceso de binarización de patrones de bloque codificados

Este proceso acepta como argumento una petición de binarización del elemento sintáctico coded\_block\_pattern.

Este proceso genera como resultado la binarización del elemento sintáctico.

La binarización del coded\_block\_pattern consiste en la concatenación de una parte prefijo y una parte sufijo. La parte prefijo de la binarización viene dada por la binarización FL de CodedBlockPatternLuma, con cMax = 15. La parte sufijo es la binarización TU de CodedBlockPatternChroma, con cMax = 2. La relación entre el valor del elemento sintáctico coded\_block\_pattern y los valores de CodedBlockPatternLuma y CodedBlockPatternChroma se especifican en la subcláusula 7.4.5.

### 9.3.2.7 Proceso de binarización para mb\_qp\_delta

Este proceso acepta como argumento una petición de binarización del elemento sintáctico mb\_qp\_delta.

Este proceso genera como resultado la binarización del elemento sintáctico.

La cadena bin de mb\_qp\_delta se calcula mediante la binarización U del valor mapeado del elemento sintáctico mb\_qp\_delta, y la regla de asignación entre el valor con signo de mb\_qp\_delta y su valor mapeado se especifica en el cuadro 9-3.

### 9.3.3 Flujo del proceso de decodificación

Este proceso acepta como argumentos una binarización del elemento sintáctico solicitado, maxBinIdxCtx, bypassFlag y ctxIdxOffset, como se describe en la subcláusula 9.3.2.

Este proceso genera como resultado el valor del elemento sintáctico.



Este proceso especifica cómo se analiza cada bit de una cadena de bits para cada elemento sintáctico.

Tras analizar cada bit, la cadena de bits resultante se compara con todas las cadenas bin de la binarización del elemento sintáctico, y se aplica lo siguiente:

- Si la cadena de bits es igual a una de las cadenas de bins, el resultado es el correspondiente valor del elemento sintáctico.
- De lo contrario (la cadena de bits no es igual a ninguna de las cadenas bins) se analiza el siguiente bit.

Al analizar cada bin, la variable binIdx se incrementa en 1, y su valor es 0 para el primer bin.

Cuando la binarización del elemento sintáctico correspondiente consta de una parte de binarización prefijo y una sufijo, la variable binIdx se pone a 0 para el primer bin de cada parte de la cadena de bins (parte prefijo o parte sufijo). En ese caso después de analizar la cadena de bits prefijo se llama al proceso de análisis de la cadena de bits sufijo que contiene las binarizaciones especificadas en las subcláusulas 9.3.2.3 y 9.3.2.5 en función de la cadena de bit prefijo resultante, según se especifica en la subcláusula 9.3.2.3 y 9.3.2.5. Obsérvese que para la binarización del elemento sintáctico coded\_block\_pattern, siempre hay una cadena de bits sufijo independientemente de la cadena de bit prefijo de longitud 4, según se especifica en la subcláusula 9.3.2.6.

En función de la variable bypassFlag, se aplica lo siguiente:

- Si bypassFlag es igual a 1, se aplica el proceso de decodificación de derivación descrito en la subcláusula 9.3.3.2.3 para analizar el valor de los bins del tren de bits.
- De lo contrario (bypassFlag es igual a 0), el análisis de cada bin se realiza en dos etapas:
  1. dados binIdx, maxBinIdxCtx y ctxIdxOffset, se calcula ctxIdx como se describe en la subcláusula 9.3.3.1;
  2. dado ctxIdx, se decodifica el valor del bin del tren de bits como se describe en la subcláusula 9.3.3.2.

### 9.3.3.1 Proceso de cálculo de ctxIdx

Este proceso acepta como argumentos binIdx, maxBinIdxCtx y ctxIdxOffset.

Este proceso genera como resultado ctxIdx.

En el cuadro 9-29 se muestra la asignación de los incrementos ctxIdx (ctxIdxInc) a binIdx para todos los valores de ctxIdxOffset excepto aquéllos relacionados con los elementos sintácticos coded\_block\_flag, significant\_coeff\_flag, last\_significant\_coeff\_flag, y coeff\_abs\_level\_minus1.

Para calcular el ctxIdx que se utilizará con un determinado binIdx, se determina en primer lugar el ctxIdxOffset asociado con la cadena bin considerada o una parte de ella. El cálculo de ctxIdx es el siguiente.

- Si el ctxIdxOffset aparece en el cuadro 9-29, el ctxIdx de un binIdx es la suma de ctxIdxOffset y ctxIdxInc, que aparecen en el cuadro 9-29. Cuando en el cuadro 9-29 aparece más de un valor de un binIdx, el proceso de asignación de ctxIdxInc para binIdx se especifica con mayor detalle en las subcláusulas que aparecen entre paréntesis en la correspondiente casilla del cuadro.
- De lo contrario (ctxIdxOffset no aparece en el cuadro 9-29) el valor de ctxIdx es la suma de los siguientes términos: ctxIdxOffset y ctxIdxBlockCatOffset(ctxBlockCat), como se especifican en el cuadro 9-30 y ctxIdxInc(ctxBlockCat). En la subcláusula 9.3.3.1.3 se especifica qué ctxBlockCat se utiliza. La subcláusula 9.3.3.1.1.9 especifica la asignación de ctxIdxInc(ctxBlockCat) para coded\_block\_flag, y la subcláusula 9.3.3.1.3 especifica la asignación de ctxIdxInc(ctxBlockCat) para significant\_coeff\_flag, last\_significant\_coeff\_flag y coeff\_abs\_level\_minus1.

Todos los bins con binIdx mayores que maxBinIdxCtx se analizan utilizando ctxIdx igual a maxBinIdxCtx.

Todos los elementos del cuadro 9-29 con "na" corresponden a valores de binIdx que no son posibles para el correspondiente ctxIdxOffset.

ctxIdx = 276 se asigna al binIdx de mb\_type que indica el modo I\_PCM. Para analizar el valor de los correspondientes bins del tren de bits, se aplica el proceso de decodificación aritmética para decisiones antes de la terminación, que se describe en la subcláusula 9.3.3.2.4.

**Cuadro 9-29 – Asignación de ctxIdxInc a binIdx para todos los valores de ctxIdxOffset excepto aquéllos relacionados con los elementos sintácticos coded\_block\_flag, significant\_coeff\_flag, last\_significant\_coeff\_flag, y oeff\_abs\_level\_minus1**

ctxIdxOffset	binIdx						
	0	1	2	3	4	5	>= 6
0	0,1,2 (subcláusula 9.3.3.1.1.3)	na	na	na	na	na	na
3	0,1,2 (subcláusula 9.3.3.1.1.3)	ctxIdx=276	3	4	5,6 (subcláusula 9.3.3.1.2)	6,7 (subcláusula 9.3.3.1.2)	7
11	0,1,2 (subcláusula 9.3.3.1.1.1)	na	na	na	na	na	na
14	0	1	2,3 (subcláusula 9.3.3.1.2)	na	na	na	na
17	0	ctxIdx=276	1	2	2,3 (subcláusula 9.3.3.1.2)	3	3
21	0	1	2	na	na	na	na
24	0,1,2 (subcláusula 9.3.3.1.1.1)	na	na	na	na	na	na
27	0,1,2 (subcláusula 9.3.3.1.1.3)	3	4,5 (subcláusula 9.3.3.1.2)	5	5	5	5
32	0	ctxIdx=276	1	2	2,3 (subcláusula 9.3.3.1.2)	3	3
36	0	1	2,3 (subcláusula 9.3.3.1.2)	3	3	3	na
40	0,1,2 (subcláusula 9.3.3.1.1.7)	3	4	5	6	6	6
47	0,1,2 (subcláusula 9.3.3.1.1.7)	3	4	5	6	6	6
54	0,1,2,3 (subcláusula 9.3.3.1.1.6)	4	5	5	5	5	5
60	0,1 (subcláusula 9.3.3.1.1.5)	2	3	3	3	3	3

**Cuadro 9-29 – Asignación de ctxIdxInc a binIdx para todos los valores de ctxIdxOffset excepto aquéllos relacionados con los elementos sintácticos coded\_block\_flag, significant\_coeff\_flag, last\_significant\_coeff\_flag, y oeff\_abs\_level\_minus1**

ctxIdxOffset	binIdx						
	0	1	2	3	4	5	>= 6
64	0,1,2 (subcláusula 9.3.3.1.1.8)	3	3	na	na	na	na
68	0	na	na	na	na	na	na
69	0	0	0	na	na	na	na
70	0,1,2 (subcláusula 9.3.3.1.1.2)	na	na	na	na	na	na
73	0,1,2,3 (subcláusula 9.3.3.1.1.4)	0,1,2,3 (subcláusula 9.3.3.1.1.4)	0,1,2,3 (subcláusula 9.3.3.1.1.4)	0,1,2,3 (subcláusula 9.3.3.1.1.4)	na	na	na
77	0,1,2,3 (subcláusula 9.3.3.1.1.4)	4,5,6,7 (subcláusula 9.3.3.1.1.4)	na	na	na	na	na
276	0	na	na	na	na	na	na

El cuadro 9-30 muestra los valores de ctxIdxBlockCatOffset en función de ctxBlockCat para los elementos sintácticos coded\_block\_flag, significant\_coeff\_flag, last\_significant\_coeff\_flag, y coeff\_abs\_level\_minus1. La especificación de ctxBlockCat se indica en el cuadro 9-32.

**Cuadro 9-30 – Asignación de ctxIdxBlockCatOffset a ctxBlockCat para los elementos sintácticos coded\_block\_flag, significant\_coeff\_flag, last\_significant\_coeff\_flag, y coeff\_abs\_level\_minus1**

Elemento sintáctico	ctxBlockCat (según se especifica en el cuadro 9-32)				
	0	1	2	3	4
coded_block_flag	0	4	8	12	16
significant_coeff_flag	0	15	29	44	47
last_significant_coeff_flag	0	15	29	44	47
coeff_abs_level_minus1	0	10	20	30	39

### 9.3.3.1.1 Proceso de asignación de ctxIdxInc utilizando elementos sintácticos adyacentes

La subcláusula 9.3.3.1.1.1 especifica el proceso de cálculo de ctxIdxInc para el elemento sintáctico mb\_skip\_flag.

La subcláusula 9.3.3.1.1.2 especifica el proceso de cálculo de ctxIdxInc para el elemento sintáctico mb\_field\_decoding\_flag.

La subcláusula 9.3.3.1.1.3 especifica el proceso de cálculo de ctxIdxInc para el elemento sintáctico mb\_type.

La subcláusula 9.3.3.1.1.4 especifica el proceso de cálculo de ctxIdxInc para el elemento sintáctico coded\_block\_pattern.

La subcláusula 9.3.3.1.1.5 especifica el proceso de cálculo de ctxIdxInc para el elemento sintáctico mb\_qp\_delta.

La subcláusula 9.3.3.1.1.6 especifica el proceso de cálculo de `ctxIdxInc` para los elementos sintácticos `ref_idx_l0` y `ref_idx_l1`.

La subcláusula 9.3.3.1.1.7 especifica el proceso de cálculo de `ctxIdxInc` para los elementos sintácticos `mvd_l0` y `mvd_l1`.

La subcláusula 9.3.3.1.1.8 especifica el proceso de cálculo de `ctxIdxInc` para el elemento sintáctico `intra_chroma_pred_mode`.

La subcláusula 9.3.3.1.1.9 especifica el proceso de cálculo de `ctxIdxInc` para el elemento sintáctico `coded_block_flag`.

#### **9.3.3.1.1 Proceso de cálculo de `ctxIdxInc` para el elemento sintáctico `mb_skip_flag`**

Este proceso produce como resultado `ctxIdxInc`.

Cuando `MbaffFrameFlag` es igual a 1 y (todavía) no se ha decodificado el `mb_field_decoding_flag` del par de macrobloques considerado cuya dirección del macrobloque superior es  $2 * (\text{CurrMbAddr} / 2)$ , se aplica la regla de cálculo para el elemento sintáctico `mb_field_decoding_flag` descrito en la subcláusula 7.4.4.

Se llama al proceso de cálculo del macrobloques adyacentes, descrito en la subcláusula 6.4.7.1, y el resultado se asigna a `mbAddrA` y `mbAddrB`.

Se calcula la variable `condTermFlagN` (siendo  $N=A$  o  $B$ ).

- Si `mbAddrN` no está disponible o `mb_skip_flag` para el macrobloque `mbAddrN` es igual a 1, `condTermFlagN` se pone a 0.
- De lo contrario (`mbAddrN` está disponible y `mb_skip_flag` para el macrobloque `mbAddrN` es igual a 0), `condTermFlagN` se pone a 1.

La variable `ctxIdxInc` se calcula del modo siguiente

$$\text{ctxIdxInc} = \text{condTermFlagA} + \text{condTermFlagB} \quad (9-1)$$

#### **9.3.3.1.1.2 Proceso de cálculo de `ctxIdxInc` para el elemento sintáctico `mb_field_decoding_flag`**

Este proceso produce como resultado `ctxIdxInc`.

Se llama al proceso de cálculo de direcciones de macrobloque adyacentes y su disponibilidad en cuadros MBAFF, descrito en la subcláusula 6.4.6, y el resultado se asigna a `mbAddrA` y `mbAddrB`.

Cuando los dos macrobloques `mbAddrN` y `mbAddrN + 1` tienen `mb_type` igual a `P_Skip` o `B_Skip`, se aplica la regla de cálculo del elemento sintáctico `mb_field_decoding_flag` descrito en la subcláusula 7.4.4 para el macrobloque `mbAddrN`.

Se calcula la variable `condTermFlagN` (siendo  $N=A$  o  $B$ ).

- Si se cumple alguna de las siguientes condiciones, `condTermFlagN` se pone a 0,
  - `mbAddrN` no está disponible
  - el macrobloque `mbAddrN` es un macrobloque cuadro.
- De lo contrario, `condTermFlagN` se pone a 1.

La variable `ctxIdxInc` se calcula del modo siguiente:

$$\text{ctxIdxInc} = \text{condTermFlagA} + \text{condTermFlagB} \quad (9-2)$$

#### **9.3.3.1.1.3 Proceso de cálculo de `ctxIdxInc` para el elemento sintáctico `mb_type`**

Este proceso acepta como argumento `ctxIdxOffset`.

Este proceso genera como resultado `ctxIdxInc`.

Se llama al proceso de cálculo de macrobloques adyacentes descrito en la subcláusula 6.4.7.1, y el resultado se asigna a mbAddrA y mbAddrB.

Se calcula la variable condTermFlagN (siendo N=A o B) como sigue.

- Si se cumple alguna de las condiciones siguientes, condTermFlagN se pone a 0.
  - mbAddrN no está disponible
  - ctxIdxOffset es igual a 0 y mb\_type del macrobloque mbAddrN es igual a SI
  - ctxIdxOffset es igual a 3 y mb\_type del macrobloque mbAddrN es igual a I\_4x4
  - ctxIdxOffset es igual a 27 y el macrobloque mbAddrN está obviado
  - ctxIdxOffset es igual a 27 y mb\_type para el macrobloque mbAddrN es igual a B\_Direct\_16x16.
- De lo contrario, condTermFlagN se pone a 1.

La variable ctxIdxInc se calcula del modo siguiente:

$$\text{ctxIdxInc} = \text{condTermFlagA} + \text{condTermFlagB} \quad (9-3)$$

#### **9.3.3.1.1.4 Proceso de cálculo de ctxIdxInc para el elemento sintáctico coded\_block\_pattern**

Este proceso acepta como argumentos ctxIdxOffset y binIdx.

Este proceso genera como resultado ctxIdxInc.

En función del valor de la variable ctxIdxOffset, se aplica lo siguiente:

- Si ctxIdxOffset es igual a 73, se aplica lo siguiente:
  - Se llama al proceso de cálculo de bloques luma 8x8 adyacentes descrito en la subcláusula 6.4.7.2, pasándole como argumento luma8x8BlkIdx = binIdx y el resultado se asigna a mbAddrA, mbAddrB, luma8x8BlkIdxA, y luma8x8BlkIdxB.
  - Se calcula la variable condTermFlagN (siendo N=A o B) como sigue.
    - Si se cumple alguna de las condiciones siguientes, condTermFlagN se pone a 0
      - mbAddrN no está disponible;
      - mb\_type del macrobloque mbAddrN es igual a I\_PCM;
      - el macrobloque mbAddrN no está obviado y  $((\text{CodedBlockPatternLuma} \gg \text{luma8x8BlkIdxN}) \& 1)$  es distinto de 0 para el macrobloque mbAddrN.
    - De lo contrario, condTermFlagN se pone a 1.
  - La variable ctxIdxInc se calcula mediante la expresión
$$\text{ctxIdxInc} = \text{condTermFlagA} + 2 * \text{condTermFlagB} \quad (9-4)$$
- De lo contrario (ctxIdxOffset es igual a 77), se aplica lo siguiente:
  - se llama al proceso de cálculo de macrobloques adyacentes descrito en la subcláusula 6.4.7.1 y el resultado se asigna a mbAddrA y mbAddrB.
- Se calcula la variable condTermFlagN (siendo N=A o B).
  - Si mbAddrN está disponible y mb\_type para el macrobloque mbAddrN es igual a I\_PCM, condTermFlagN se pone a 1.
  - De lo contrario si se cumple alguna de las condiciones siguientes, condTermFlagN se pone a 0.
    - mbAddrN no está disponible o el macrobloque mbAddrN está obviado

- binIdx es igual a 0 y CodedBlockPatternChroma para el macrobloque mbAddrN es distinto de 0
- binIdx es igual a 1 y CodedBlockPatternChroma para el macrobloque mbAddrN es distinto de 2
- De lo contrario, condTermFlagN se pone a 1.
- Se calcula la variable ctxIdxInc mediante la expresión:

$$\text{ctxIdxInc} = \text{condTermFlagA} + 2 * \text{condTermFlagB} + ((\text{binIdx} == 1) ? 4 : 0) \quad (9-5)$$

NOTA – Cuando el macrobloque utiliza el modo de predicción Intra\_16x16, los valores de CodedBlockPatternLuma y CodedBlockPatternChroma para el macrobloque se calcula a partir de mb\_type, como se describe en el cuadro 7-8.

### 9.3.3.1.1.5 Proceso de cálculo de ctxIdxInc para el elemento sintáctico mb\_qp\_delta

Este proceso genera como resultado ctxIdxInc.

Sea prevMbAddr la dirección macrobloque del macrobloque que precede al macrobloque considerado en orden de decodificación. Cuando el macrobloque considerado es el primer macrobloque de un sector, prevMbAddr se marca como no disponible.

Se calcula la variable ctxIdxInc.

- Si se cumple alguna de las siguientes condiciones ctxIdxInc es igual a 0.
  - prevMbAddr no está disponible o el macrobloque prevMbAddr está obviado
  - mb\_type del macrobloque prevMbAddr es igual a I\_PCM
  - El macrobloque prevMbAddr no está codificado en modo de predicción Intra\_16x16 y tanto CodedBlockPatternLuma como CodedBlockPatternChroma del macrobloque prevMbAddr son iguales a 0
  - mb\_qp\_delta del macrobloque prevMbAddr es igual a 0.
- De lo contrario, ctxIdxInc se pone a 1.

### 9.3.3.1.1.6 Proceso de cálculo de ctxIdxInc para elementos sintácticos ref\_idx\_10 y ref\_idx\_11

Este proceso acepta como argumentos mbPartIdx y el sufijo de la lista de imágenes de referencia IX, siendo X = 0 ó 1.

Este proceso genera como resultado ctxIdxInc.

Fijese currSubMbType igual a sub\_mb\_type[ mbPartIdx ].

Se invoca el proceso de cálculo de particiones adyacentes descrito en la subcláusula 6.4.7.5, pasándole como argumento mbPartIdx, currSubMbType, y subMbPartIdx = 0, y el resultado se asigna a mbAddrA\mbPartIdxA y mbAddrB\mbPartIdxB.

Se calcula la variable refIdxZeroFlagN, con ref\_idx\_IX[ mbPartIdxN ] (siendo N=A o B) especifica el elemento sintáctico del macrobloque mbAddrN.

- Si MbaffFrameFlag es igual a 1, el macrobloque considerado es un macrobloque cuadro y el macrobloque mbAddrN es un macrobloque campo.

$$\text{refIdxZeroFlagN} = ((\text{ref\_idx\_IX}[\text{mbPartIdxN}] > 1) ? 0 : 1) \quad (9-6)$$

- De lo contrario,

$$\text{refIdxZeroFlagN} = ((\text{ref\_idx\_IX}[\text{mbPartIdxN}] > 0) ? 0 : 1) \quad (9-7)$$

Se calcula la variable predModeEqualFlag como sigue:

- Si el macrobloque mbAddrN tiene mb\_type igual a P\_8x8 o B\_8x8, se aplica lo siguiente:

- Si  $\text{SubMbPredMode}(\text{sub\_mb\_type}[\text{mbPartIdxN}])$  es distinto de  $\text{Pred\_LX}$  y es distinto de  $\text{BiPred}$ ,  $\text{predModeEqualFlag}$  se pone a 0, donde  $\text{sub\_mb\_type}$  especifica el elemento sintáctico del macrobloque  $\text{mbAddrN}$ .
- De lo contrario,  $\text{predModeEqualFlag}$  se pone a 1.
- De lo contrario, se aplica lo siguiente:
  - Si  $\text{MbPartPredMode}(\text{mb\_type}, \text{mbPartIdxN})$  es distinto de  $\text{Pred\_LX}$  y es distinto de  $\text{BiPred}$ ,  $\text{predModeEqualFlag}$  se pone a 0, donde  $\text{mb\_type}$  especifica el elemento sintáctico del macrobloque  $\text{mbAddrN}$ .
  - De lo contrario,  $\text{predModeEqualFlag}$  se pone a 1.

Se calcula la variable  $\text{condTermFlagN}$  (siendo  $N=A$  o  $B$ ) como sigue:

- Si se cumple alguna de las siguientes condiciones,  $\text{condTermFlagN}$  se pone a 0.
  - $\text{mbAddrN}$  no está disponible;
  - el macrobloque  $\text{mbAddrN}$  tiene  $\text{mb\_type}$  igual a  $\text{P\_Skip}$  o  $\text{B\_Skip}$ ;
  - el macrobloque  $\text{mbAddrN}$  está codificado con el modo predicción intra;
  - $\text{predModeEqualFlag}$  es igual a 0;
  - $\text{refIdxZeroFlagN}$  es igual a 1.
- De lo contrario,  $\text{condTermFlagN}$  es igual a 1.

La variable  $\text{ctxIdxInc}$  se calcula mediante la expresión:

$$\text{ctxIdxInc} = \text{condTermFlagA} + 2 * \text{condTermFlagB} \quad (9-8)$$

#### 9.3.3.1.1.7 Proceso de cálculo de $\text{ctxIdxInc}$ para elementos sintácticos $\text{mvd}_{10}$ y $\text{mvd}_{11}$

Este proceso acepta como argumentos  $\text{mbPartIdx}$ ,  $\text{subMbPartIdx}$ , el sufijo de la lista de imágenes de referencia  $\text{IX}$ , y  $\text{ctxIdxOffset}$ .

Este proceso genera como resultado  $\text{ctxIdxInc}$ .

Fijese  $\text{currSubMbType}$  igual a  $\text{sub\_mb\_type}[\text{mbPartIdx}]$ .

Se llama al proceso de cálculo de particiones adyacentes descrito en la subcláusula 6.4.7.5, pasándole como argumentos  $\text{mbPartIdx}$ ,  $\text{currSubMbType}$ , y  $\text{subMbPartIdx}$ , y el resultado se asigna a  $\text{mbAddrA}\backslash\text{mbPartIdxA}\backslash\text{subMbPartIdxA}$  y  $\text{mbAddrB}\backslash\text{mbPartIdxB}\backslash\text{subMbPartIdxB}$ .

Se calcula la variable  $\text{compIdx}$  como sigue:

- Si  $\text{ctxIdxOffset}$  es igual a 40,  $\text{compIdx}$  se pone a 0.
- De lo contrario ( $\text{ctxIdxOffset}$  es igual a 47),  $\text{compIdx}$  se pone a 1.

Se calcula la variable  $\text{predModeEqualFlag}$  como sigue:

- Si el macrobloque  $\text{mbAddrN}$  tiene  $\text{mb\_type}$  igual a  $\text{P\_8x8}$  o  $\text{B\_8x8}$ , se aplica lo siguiente:
  - Si  $\text{SubMbPredMode}(\text{sub\_mb\_type}[\text{mbPartIdxN}])$  es distinto de  $\text{Pred\_LX}$  y es distinto de  $\text{BiPred}$ ,  $\text{predModeEqualFlag}$  se pone a 0, donde  $\text{sub\_mb\_type}$  especifica el elemento sintáctico del macrobloque  $\text{mbAddrN}$ .
  - De lo contrario,  $\text{predModeEqualFlag}$  se pone a 1.
- De lo contrario, se aplica lo siguiente:
  - Si  $\text{MbPartPredMode}(\text{mb\_type}, \text{mbPartIdxN})$  es distinto de  $\text{Pred\_LX}$  y es distinto de  $\text{BiPred}$ ,  $\text{predModeEqualFlag}$  se pone a 0, donde  $\text{mb\_type}$  especifica el elemento sintáctico del macrobloque  $\text{mbAddrN}$ .
  - De lo contrario,  $\text{predModeEqualFlag}$  se pone a 1.

Se calcula la variable  $\text{absMvdCompN}$  (siendo  $N=A$  o  $B$ ) como sigue:

- Si se cumple alguna de las siguientes condiciones, `absMvdCompN` se pone a 0.
  - `mbAddrN` no está disponible
  - el macrobloque `mbAddrN` tiene `mb_type` igual a `P_Skip` o `B_Skip`;
  - el macrobloque `mbAddrN` está codificado en modo predicción intra;
  - `predModeEqualFlag` es igual a 0.
- De lo contrario, se aplica lo siguiente:
  - Si `compIdx` es igual a 1, `MbaffFrameFlag` es igual a 1, el macrobloque considerado es un macrobloque cuadro y el macrobloque `mbAddrN` es un macrobloque campo
 
$$\text{absMvdCompN} = \text{Abs}(\text{mvd\_IX}[\text{mbPartIdxN}][[\text{subMbPartIdxN}][[\text{compIdx}]]) * 2 \quad (9-9)$$
  - De lo contrario, si `compIdx` es igual a 1, `MbaffFrameFlag` es igual a 1, el macrobloque considerado es un macrobloque campo, y el macrobloque `mbAddrN` es un macrobloque cuadro
 
$$\text{absMvdCompN} = \text{Abs}(\text{mvd\_IX}[\text{mbPartIdxN}][[\text{subMbPartIdxN}][[\text{compIdx}]]) / 2 \quad (9-10)$$
  - De lo contrario,
 
$$\text{absMvdCompN} = \text{Abs}(\text{mvd\_IX}[\text{mbPartIdxN}][[\text{subMbPartIdxN}][[\text{compIdx}]]) \quad (9-11)$$

La variable `ctxIdxInc` se calcula del modo siguiente:

- Si ( `absMvdCompA + absMvdCompB` ) es menor que 3, `ctxIdxInc` se pone a 0.
- De lo contrario, si ( `absMvdCompA + absMvdCompB` ) es mayor que 32, `ctxIdxInc` se pone a 2.
- De lo contrario ( ( `absMvdCompA + absMvdCompB` ) está entre 3 y 32, inclusive), `ctxIdxInc` se pone a 1.

#### **9.3.3.1.1.8 Proceso de cálculo de `ctxIdxInc` para el elemento sintáctico `intra_chroma_pred_mode`**

Este proceso genera como resultado `ctxIdxInc`.

Se llama al proceso de cálculo de macrobloques adyacentes descrito en la subcláusula 6.4.7.1 y el resultado se asigna a `mbAddrA` y `mbAddrB`.

Se calcula la variable `condTermFlagN` (siendo `N=A` o `B`) como sigue:

- Si se cumple una de las siguientes condiciones, `condTermFlagN` se pone a 0
  - `mbAddrN` no está disponible;
  - el macrobloque `mbAddrN` está codificado en modo predicción intra;
  - `mb_type` para el macrobloque `mbAddrN` es igual a `I_PCM`;
  - `intra_chroma_pred_mode` del macrobloque `mbAddrN` es igual a 0.
- De lo contrario, `condTermFlagN` se pone a 1.

La variable `ctxIdxInc` se calcula mediante la expresión:

$$\text{ctxIdxInc} = \text{condTermFlagA} + \text{condTermFlagB} \quad (9-12)$$

#### **9.3.3.1.1.9 Proceso de cálculo de `ctxIdxInc` para el elemento sintáctico `coded_block_flag`**

Este proceso acepta como argumento `ctxBlockCat` y otros argumentos como se indica a continuación:

- si `ctxBlockCat` es igual a 0, no se le pasan otros argumentos;
- de lo contrario, si `ctxBlockCat` es igual a 1 ó 2, se le pasa el argumento `luma4x4BlkIdx`;
- de lo contrario, si `ctxBlockCat` es igual a 3, se le pasa como argumento el índice de componente `chroma iCbCr`;



- de lo contrario (ctxBlockCat es igual a 4), se le pasan como argumentos chroma4x4BlkIdx y el índice de componente croma compIdx.

Este proceso genera como resultado ctxIdxInc( ctxBlockCat ).

Se calcula la variable transBlockN (siendo N=A o B) como sigue:

- Si ctxBlockCat es igual a 0, se aplica lo siguiente:
  - Se llama al proceso de cálculo de macrobloques adyacentes descrito en la subcláusula 6.4.7.1 y el resultado se asigna a mbAddrN (siendo N=A o B).
  - Se calcula la variable transBlockN como sigue:
    - Si mbAddrN está disponible y el macrobloque mbAddrN está codificado en modo predicción Intra\_16x16, transBlockN será igual al bloque c.c. luma del macrobloque mbAddrN.
    - De lo contrario, transBlockN se marca como no disponible.
- De lo contrario, si ctxBlockCat es igual a 1 ó 2, se aplica lo siguiente.
  - Se llama al proceso de cálculo de bloques luma 4x4 adyacentes descrito en la subcláusula 6.4.7.3 pasándole como argumento luma4x4BlkIdx y el resultado se asigna a mbAddrN, luma4x4BlkIdxN (siendo N=A o B).
  - Se calcula la variable transBlockN como sigue:
    - Si mbAddrN está disponible, el macrobloque mbAddrN no está obviado, mb\_type del macrobloque mbAddrN es distinto de I\_PCM, y  $((\text{CodedBlockPatternLuma} \gg (\text{luma4x4BlkIdxN} \gg 2)) \& 1)$  es distinto de 0 para el macrobloque mbAddrN, se asigna a transBlockN el bloque luma 4x4 con luma4x4BlkIdxN del macrobloque mbAddrN.
    - De lo contrario, transBlockN se marca como no disponible.
- De lo contrario, si ctxBlockCat es igual a 3, se aplica lo siguiente:
  - Se llama al proceso de cálculo de macrobloques adyacentes descrito en la subcláusula 6.4.7.1 y el resultado se asigna a mbAddrN (siendo N=A o B).
  - Se calcula la variable transBlockN como sigue:
    - Si mbAddrN está disponible, el macrobloque mbAddrN no está obviado, mb\_type del macrobloque mbAddrN es distinto de I\_PCM, y CodedBlockPatternChroma es distinto de 0 para el macrobloque mbAddrN, se asigna a transBlockN el bloque de DC croma de la componente croma iCbCr del macrobloque mbAddrN.
    - De lo contrario, transBlockN se marca como no disponible.
- De lo contrario (ctxBlockCat es igual a 4), se aplica lo siguiente:
  - Se llama al proceso de cálculo de bloques croma 4x4 descrito en la subcláusula 6.4.7.4, pasándole como argumento chroma4x4BlkIdx, y el resultado se asigna a mbAddrN, chroma4x4BlkIdxN (siendo N=A o B).
  - Se calcula la variable transBlockN como sigue:
    - Si mbAddrN está disponible, el macrobloque mbAddrN no está obviado, mb\_type del macrobloque mbAddrN es distinto de I\_PCM, y CodedBlockPatternChroma es igual a 2 para el macrobloque mbAddrN, se asigna a transBlockN el bloque croma 4x4 con chroma4x4BlkIdxN de la componente croma iCbCr del macrobloque mbAddrN.
    - De lo contrario, transBlockN se marca como no disponible.

Se calcula la variable condTermFlagN (siendo N=A o B) como sigue:

- Si se cumple alguna de las siguientes condiciones, condTermFlagN se pone a 0.

- mbAddrN no está disponible y el macrobloque considerado está codificado con el modo predicción intra;
- mbAddrN está disponible y transBlockN no está disponible y el mb\_type del macrobloque mbAddrN es distinto de I\_PCM;
- el macrobloque considerado está codificado en modo predicción intra, constrained\_intra\_pred\_flag es igual a 1, el macrobloque mbAddrN está disponible y codificado en modo predicción intra, y se utilizan particiones de datos de sector (nal\_unit\_type está entre 2 y 4, inclusive).
- De lo contrario, si se cumple alguna de las siguientes condiciones, condTermFlagN se pone a 1
  - mbAddrN no está disponible y el macrobloque considerado está codificado en modo predicción intra;
  - mb\_type del macrobloque mbAddrN es igual a I\_PCM.
- De lo contrario, se asigna a condTermFlagN el valor del coded\_block\_flag del bloque de transformada transBlockN que se codificó para el macrobloque mbAddrN.

La variable ctxIdxInc( ctxBlockCat ) se calcula mediante la expresión:

$$\text{ctxIdxInc}(\text{ctxBlockCat}) = \text{condTermFlagA} + 2 * \text{condTermFlagB} \quad (9-13)$$

### 9.3.3.1.2 Proceso de asignación de ctxIdxInc utilizando valores bins decodificados previamente

Este proceso acepta como argumentos ctxIdxOffset y binIdx.

Este proceso genera como resultado ctxIdxInc.

El cuadro 9-31 contiene la especificación de ctxIdxInc en función de los valores ctxIdxOffset y binIdx.

Para cada valor de ctxIdxOffset y binIdx, ctxIdxInc se calcula utilizando algunos de los valores bin decodificados previamente (  $b_0, b_1, b_2, \dots, b_k$  ), donde el valor del índice k es menor que el valor de binIdx.

**Cuadro 9-31 – Especificación de ctxIdxInc en función de los valores de ctxIdxOffset y binIdx**

Valor (nombre) de ctxIdxOffset	binIdx	ctxIdxInc
3	4	$(b_3 \neq 0) ? 5 : 6$
	5	$(b_3 \neq 0) ? 6 : 7$
14	2	$(b_1 \neq 1) ? 2 : 3$
17	4	$(b_3 \neq 0) ? 2 : 3$
27	2	$(b_1 \neq 0) ? 4 : 5$
32	4	$(b_3 \neq 0) ? 2 : 3$
36	2	$(b_1 \neq 0) ? 2 : 3$

### 9.3.3.1.3 Proceso de asignación de ctxIdxInc para los elementos sintácticos significant\_coeff\_flag, last\_significant\_coeff\_flag y coeff\_abs\_level\_minus1

Este proceso acepta como argumentos ctxIdxOffset y binIdx.

Este proceso produce como resultado ctxIdxInc.

El proceso de asignación de ctxIdxInc para los elementos sintácticos significant\_coeff\_flag, last\_significant\_coeff\_flag y coeff\_abs\_level\_minus1, así como para coded\_block\_flag, depende de

las categorías de los diferentes bloques indicados mediante la variable ctxBlockCat. En el cuadro 9-32 se especifican estas categorías de bloque.

**Cuadro 9-32 – Especificación de ctxBlockCat para los diferentes bloques**

Descripción del bloque	maxNumCoeff	ctxBlockCat
Bloque de niveles de coeficientes de transformada c.c. luma (para macrobloques codificados en modo predicción Intra_16x16)	16	0
Bloque de niveles de coeficientes de transformada c.a. luma (para macrobloques codificados en modo predicción Intra_16x16)	15	1
Bloque de niveles de coeficientes de transformada luma (para macrobloques no codificados en modo predicción Intra_16x16)	16	2
Bloque de niveles de coeficientes de transformada c.c. croma	4	3
Bloque de niveles de coeficientes de transformada c.a. croma	15	4

Para los elementos sintácticos significant\_coeff\_flag y last\_significant\_coeff\_flag se asigna a ctxIdxInc la posición de barrido scanningPos del bloque considerado, donde scanningPos toma valores entre 0 y maxNumCoeff – 2, inclusive:

$$\text{ctxIdxInc} = \text{scanningPos} \quad (9-14)$$

La posición de barrido de bloques cuadro codificados corresponde al barrido en zig-zag; la posición de barrido de bloques campo codificados corresponde al barrido por campo.

Sea numDecodAbsLevelEq1 la suma del número de niveles de coeficientes de transformada decodificados con valor absoluto igual a 1, y sea numDecodAbsLevelGt1 la suma del número de niveles de coeficiente de transformada decodificados con valor absoluto mayor que 1. Cuando se aplica el proceso de decodificación, los dos números se refieren al mismo bloque de coeficiente de transformada. Para decodificar coeff\_abs\_level\_minus1, se calcula el valor de ctxIdxInc para coeff\_abs\_level\_minus1 en función de binIdx como se describe a continuación.

– Si binIdx es igual a 0, ctxIdxInc se calcula mediante la expresión:

$$\text{ctxIdxInc} = ((\text{numDecodAbsLevelGt1} \neq 0) ? 0 : \text{Min}(4, 1 + \text{numDecodAbsLevelEq1})) \quad (9-15)$$

– De lo contrario (binIdx es mayor que 0), ctxIdxInc se calcula mediante la expresión:

$$\text{ctxIdxInc} = 5 + \text{Min}(4, \text{numDecodAbsLevelGt1}) \quad (9-16)$$

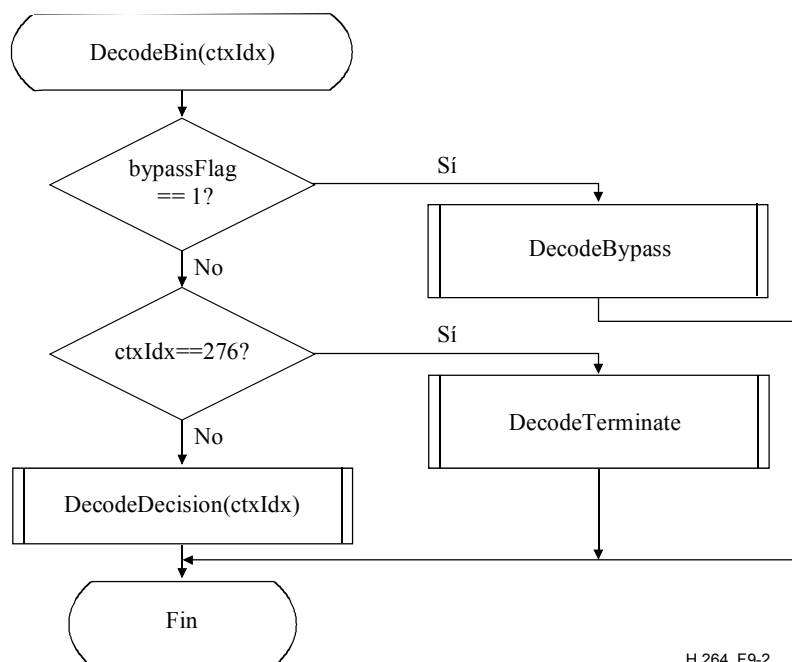
### 9.3.3.2 Proceso de decodificación aritmética

Este proceso acepta como argumentos bypassFlag, ctxIdx, calculado como se describe en la subcláusula 9.3.3.1, y las variables de estado codIRange y codIOffset del mecanismo de decodificación aritmética.

Este proceso genera como resultado el valor del bin.

La figura 9-2 ilustra todo el proceso de decodificación aritmética de un solo bin. Para decodificar el valor del bin, se pasa el índice de contexto ctxIdx al proceso de decodificación aritmética DecodeBin(ctxIdx), que se describe a continuación.

- Si bypassFlag es igual a 1, se llama a DecodeBypass() descrito en la subcláusula 9.3.3.2.3.
- De lo contrario, si bypassFlag es igual a 0 y ctxIdx es igual a 276, se llama a DecodeTerminate() descrito en la subcláusula 9.3.3.2.4.
- De lo contrario (bypassFlag es igual a 0 y ctxIdx es distinto de 276), se llama a DecodeDecision() descrito en la subcláusula 9.3.3.2.1.



**Figura 9-2 – Descripción del proceso de decodificación aritmética de un solo bin (informativo)**

NOTA – La codificación aritmética se basa en el principio de subdivisión de intervalos recursiva. Dada una estimación de probabilidad  $p(0)$  y  $p(1) = 1 - p(0)$  de una decisión binaria  $(0, 1)$ , el subintervalo de codificación dado inicialmente en la gama  $\text{codIRange}$  se subdivide en dos subintervalos cuyas gamas son, respectivamente,  $p(0) * \text{codIRange}$  y  $\text{codIRange} - p(0) * \text{codIRange}$ . En función de la decisión, la cual se ha observado, el correspondiente subintervalo se elige como el nuevo intervalo de codificación, y la cadena de codificación binaria que apunta a ese intervalo representa la secuencia de decisiones binarias observadas. Es útil hacer la diferencia entre el símbolo más probable (MPS) y el símbolo menos probable (LPS), de manera que las decisiones binarias se identifican como MPS o LPS en lugar de cómo 0 ó 1. Habida cuenta de esta terminología, cada contexto se especifica mediante la probabilidad  $p_{\text{LPS}}$  del LPS y el valor de MPS ( $\text{valMPS}$ ), que es 0 ó 1.

El mecanismo básico de codificación aritmética en esta Recomendación | Norma Internacional tiene tres propiedades distintas:

- La estimación de probabilidad se calcula mediante una máquina de estados finitos con un proceso de transición basado en tablas entre 64 estados de probabilidad representativos diferentes  $\{p_{\text{LPS}}(p\text{StateIdx}) \mid 0 \leq p\text{StateIdx} < 64\}$  para la probabilidad LPS,  $p_{\text{LPS}}$ . El número de estados se elige de modo que el estado de probabilidad con índice de  $p\text{StateIdx} = 0$  corresponda al valor de probabilidad LPS de 0,5, y de manera que la probabilidad LPS disminuya al aumentar el índice de estado.
- La gama  $\text{codIRange}$  que representa el estado del mecanismo de codificación se cuantifica en un conjunto pequeño  $\{Q_1, \dots, Q_4\}$  de valores de cuantificación preestablecidos antes de calcular la nueva gama de intervalo. Al almacenar en una tabla que contiene todos los  $64 \times 4$  valores de productos precalculados de  $Q_i * p_{\text{LPS}}(p\text{StateIdx})$  se puede obtener una estimación aproximada del producto  $\text{codIRange} * p_{\text{LPS}}(p\text{StateIdx})$  sin tener que hacer la multiplicación.
- Para los elementos sintácticos o parte de éstos que se supone tienen una distribución de probabilidad aproximadamente uniforme, se utiliza un proceso de derivación de codificación y decodificación simplificado separado.

### 9.3.3.2.1 Proceso de decodificación aritmética de una decisión binaria

Este proceso acepta como argumentos  $\text{ctxIdx}$ ,  $\text{codIRange}$  y  $\text{codIOffset}$ .

Este proceso genera como resultado el valor decodificado de binVal, y las variables actualizadas codIRange y codIOffset.

En la figura 9-3 se muestra el diagrama de flujo de la decodificación de una sola decisión (DecodeDecision).

1. Se calcula el valor de la variable codIRangeLPS.
  - La variable codIRange se calcula a partir del valor actual de qCodIRangeIdx mediante la expresión:
$$qCodIRangeIdx = (codIRange \gg 6) \& 0x03 \quad (9-17)$$
  - Dados qCodIRangeIdx y pStateIdx correspondientes a ctxIdx, se asigna a codIRangeLPS el valor de la variable rangeTabLPS especificado en el cuadro 9-33:
$$codIRangeLPS = rangeTabLPS[pStateIdx][qCodIRangeIdx] \quad (9-18)$$
2. Se asigna a la variable codIRange el valor codIRange – codIRangeLPS, y se aplica lo siguiente:
  - Si codIOffset es mayor o igual que codIRange, la variable binVal se pone a 1 – valMPS, codIOffset se disminuye en codIRange, y codIRange se pone igual a codIRangeLPS.
  - De lo contrario, la variable binVal se pone a valMPS.

Dado el valor de la variable binVal, la transición de estado se realiza como se describe en la subcláusula 9.3.3.2.1.1. En función del valor actual de codIRange, se aplica una renormalización como se describe en la subcláusula 9.3.3.2.2.

#### 9.3.3.2.1.1 Proceso de transición de estado

Este proceso acepta como argumentos el pStateIdx actual, el valor decodificado binVal y los valores valMPS de la variable contexto correspondientes a ctxIdx.

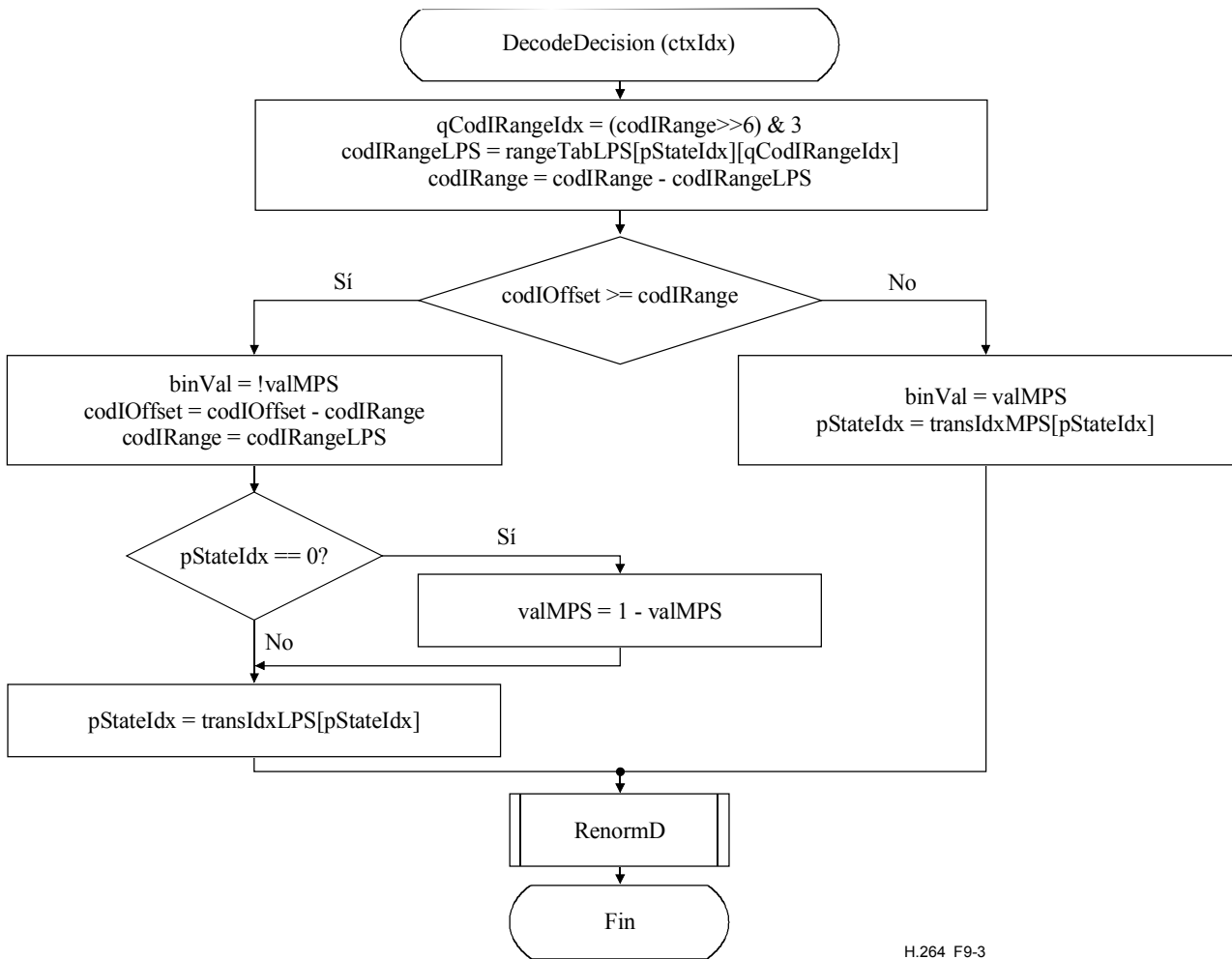
Este proceso genera como resultado el pStateIdx actualizado y valMPS de la variable contexto correspondientes a ctxIdx.

En función del valor decodificado binVal, se calcula el valor actualizado de las dos variables pStateIdx y valMPS asociados con ctxIdx mediante la siguiente expresión:

```
if( binVal == valMPS )
    pStateIdx = transIdxMPS( pStateIdx )
else {
    if( pStateIdx == 0 )
        valMPS = 1 - valMPS
    pStateIdx = transIdxLPS( pStateIdx )
}
```

(9-19)

El cuadro 9-34 especifica las reglas de transición transIdxMPS() y transIdxLPS() después de decodificar el valor de valMPS y 1 – valMPS, respectivamente.



H.264\_F9-3

**Figura 9-3 – Diagrama de flujo de la decodificación de una decisión**

**Cuadro 9-33 – Especificación de rangeTabLPS en función de pStateIdx y de qCodIRangeIdx**

pStateIdx	qCodIRangeIdx				pStateIdx	qCodIRangeIdx			
	0	1	2	3		0	1	2	3
0	128	176	208	240	32	27	33	39	45
1	128	167	197	227	33	26	31	37	43
2	128	158	187	216	34	24	30	35	41
3	123	150	178	205	35	23	28	33	39
4	116	142	169	195	36	22	27	32	37
5	111	135	160	185	37	21	26	30	35
6	105	128	152	175	38	20	24	29	33
7	100	122	144	166	39	19	23	27	31
8	95	116	137	158	40	18	22	26	30
9	90	110	130	150	41	17	21	25	28
10	85	104	123	142	42	16	20	23	27
11	81	99	117	135	43	15	19	22	25
12	77	94	111	128	44	14	18	21	24
13	73	89	105	122	45	14	17	20	23
14	69	85	100	116	46	13	16	19	22
15	66	80	95	110	47	12	15	18	21
16	62	76	90	104	48	12	14	17	20
17	59	72	86	99	49	11	14	16	19
18	56	69	81	94	50	11	13	15	18
19	53	65	77	89	51	10	12	15	17
20	51	62	73	85	52	10	12	14	16
21	48	59	69	80	53	9	11	13	15
22	46	56	66	76	54	9	11	12	14
23	43	53	63	72	55	8	10	12	14
24	41	50	59	69	56	8	9	11	13
25	39	48	56	65	57	7	9	11	12
26	37	45	54	62	58	7	9	10	12
27	35	43	51	59	59	7	8	10	11
28	33	41	48	56	60	6	8	9	11
29	32	39	46	53	61	6	7	9	10
30	30	37	43	50	62	6	7	8	9
31	29	35	41	48	63	2	2	2	2

**Cuadro 9-34 – Tabla de transiciones de estado**

<b>pStateIdx</b>	<b>0</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>6</b>	<b>7</b>	<b>8</b>	<b>9</b>	<b>10</b>	<b>11</b>	<b>12</b>	<b>13</b>	<b>14</b>	<b>15</b>
<b>transIdxLPS</b>	0	0	1	2	2	4	4	5	6	7	8	9	9	11	11	12
<b>transIdxMPS</b>	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
<b>pStateIdx</b>	<b>16</b>	<b>17</b>	<b>18</b>	<b>19</b>	<b>20</b>	<b>21</b>	<b>22</b>	<b>23</b>	<b>24</b>	<b>25</b>	<b>26</b>	<b>27</b>	<b>28</b>	<b>29</b>	<b>30</b>	<b>31</b>
<b>transIdxLPS</b>	13	13	15	15	16	16	18	18	19	19	21	21	22	22	23	24
<b>transIdxMPS</b>	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
<b>pStateIdx</b>	<b>32</b>	<b>33</b>	<b>34</b>	<b>35</b>	<b>36</b>	<b>37</b>	<b>38</b>	<b>39</b>	<b>40</b>	<b>41</b>	<b>42</b>	<b>43</b>	<b>44</b>	<b>45</b>	<b>46</b>	<b>47</b>
<b>transIdxLPS</b>	24	25	26	26	27	27	28	29	29	30	30	30	31	32	32	33
<b>transIdxMPS</b>	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
<b>pStateIdx</b>	<b>48</b>	<b>49</b>	<b>50</b>	<b>51</b>	<b>52</b>	<b>53</b>	<b>54</b>	<b>55</b>	<b>56</b>	<b>57</b>	<b>58</b>	<b>59</b>	<b>60</b>	<b>61</b>	<b>62</b>	<b>63</b>
<b>transIdxLPS</b>	33	33	34	34	35	35	35	36	36	36	37	37	37	38	38	63
<b>transIdxMPS</b>	49	50	51	52	53	54	55	56	57	58	59	60	61	62	62	63

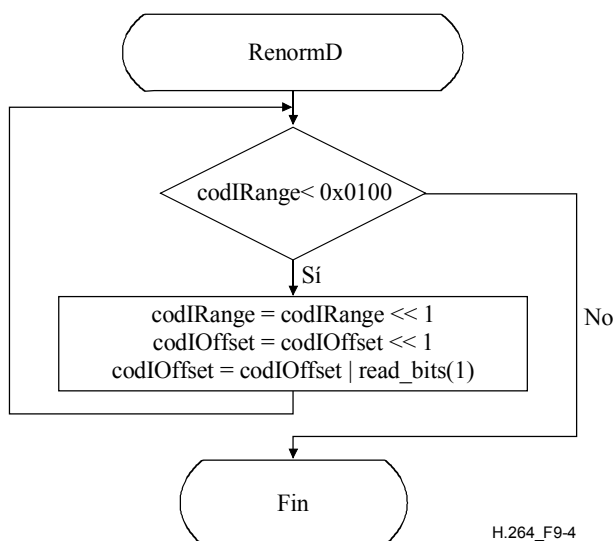
### 9.3.3.2.2 Proceso de renormalización en el mecanismo de decodificación aritmética

Este proceso acepta como argumentos los bits de datos de sector y las variables codIRange y codIOffset.

Ese proceso genera como resultado las variables actualizadas codIRange y codIOffset.

En la figura 9-4 se muestra el diagrama de flujo de la renormalización. El valor actual de codIRange se compara en primer lugar con 0x0100 y a continuación se realizan los siguientes pasos:

- Si codIRange es mayor o igual que 0x0100, la renormalización no es necesaria y termina el proceso RenormD.
- De lo contrario (codIRange es menor que 0x0100), comienza el bucle de renormalización. Dentro de este bucle, el valor codIRange se multiplica por dos, es decir, se desplaza una posición hacia la izquierda, y se utiliza la función read\_bits( 1 ) para desplazar 1 bit en codIOffset.



H.264\_F9-4

**Figura 9-4 – Diagrama de flujo de la renormalización**



### 9.3.3.2.3 Proceso de decodificación de derivación para decisiones binarias

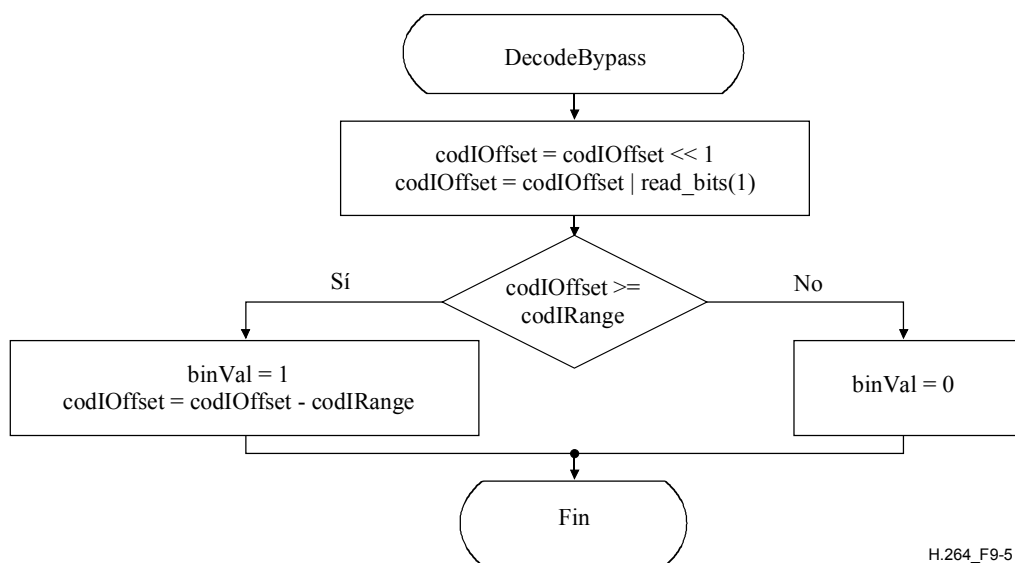
Este proceso acepta como argumentos los bits de datos de sector y las variables `codIRange` y `codIOffset`.

Este proceso genera como resultado las variables `codIRange` y `codIOffset` actualizadas, y el valor decodificado de `binVal`.

Se llama al proceso de decodificación de derivación cuando `bypassFlag` es igual a 1. En la figura 9-5 se muestra el diagrama de flujo de este proceso.

En primer lugar, el valor de `codIOffset` se multiplica por dos, es decir, se desplaza hacia la izquierda en una posición y se utiliza la función `read_bits( 1 )` para desplazar un solo bit en `codIOffset`. A continuación se compara el valor de `codIOffset` con el valor de `codIRange` y se aplica lo siguiente:

- Si `codIOffset` es mayor o igual que `codIRange`, la variable `binVal` se pone a 1 y `codIOffset` se disminuye en `codIRange`.
- De lo contrario (`codIOffset` es mayor que `codIRange`), la variable `binVal` se pone a 0.



H.264\_F9-5

Figura 9-5 – Diagrama de flujo del proceso de codificación de derivación

### 9.3.3.2.4 Proceso de decodificación de decisiones binarias antes de la terminación

Este proceso acepta como argumentos los bits de datos de sector y las variables `codIRange` y `codIOffset`.

Este proceso genera como resultado las variables `codIRange` y `codIOffset` actualizadas, y el valor decodificado `binVal`.

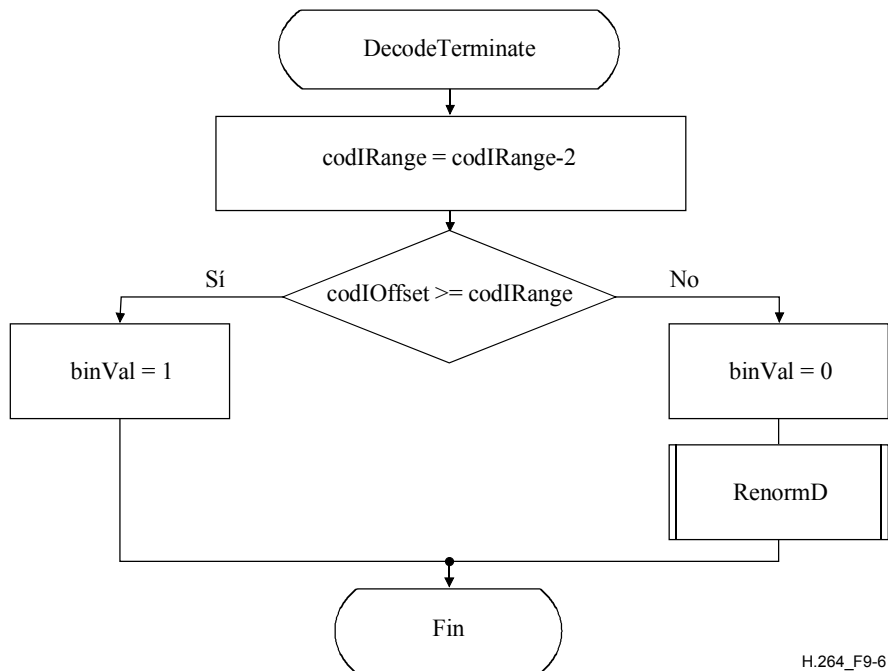
Esta rutina de decodificación especial se aplica para decodificar `end_of_slice_flag` y el bin que indica el modo `I_PCM` correspondiente a `ctxIdx` igual a 276. En la figura 9-6 se muestra el diagrama de flujo del correspondiente proceso de decodificación, que se describe a continuación.

En primer lugar, el valor de `codIRange` se disminuye en 2. Seguidamente, el valor de `codIOffset` se compara con el valor de `codIRange` y se aplica lo siguiente:

- Si `codIOffset` es mayor o igual que `codIRange`, la variable `binVal` se pone a 1, la renormalización no es necesaria y se termina la decodificación CABAC. El último bit insertado en el registro `codIOffset` es igual a 1. Al decodificar `end_of_slice_flag`, este último bit insertado en el registro `codIOffset` se interpreta como `rbsp_stop_one_bit`.

- De lo contrario ( $\text{codIOffset}$  es menor que  $\text{codIRange}$ ), la variable  $\text{binVal}$  se pone a 0 y se aplica la renormalización como se describe en la subcláusula 9.3.3.2.2.

NOTA – Este procedimiento también se puede implementar mediante  $\text{DecodeDecision}(\text{ctxIdx})$ , con  $\text{ctxIdx} = 276$ . Cuando el valor decodificado es igual a 1,  $\text{DecodeDecision}(\text{ctxIdx})$  lee siete bits más y en consecuencia el proceso de decodificación tiene que ajustar su puntero al tren de bits para decodificar correctamente los siguientes elementos sintácticos.



H.264\_F9-6

**Figura 9-6 – Diagrama de flujo de la decodificación de una decisión antes de la terminación**

### 9.3.4 Proceso de codificación aritmética (informativo)

Esta subcláusula no es parte integrante de esta Recomendación | Norma Internacional.

Este proceso acepta como argumentos decisiones que se tienen que codificar y escribir.

Este proceso genera como resultado los bits que se han de escribir en la RBSP.

En esta subcláusula informativa se describe el mecanismo de codificación aritmética que corresponde al mecanismo de decodificación aritmética descrito en la subcláusula 9.3.3.2. El mecanismo de codificación es esencialmente simétrico al de decodificación, es decir, el orden de llamada a los procedimientos es el mismo. En esta cláusula se describen los siguientes procedimientos:  $\text{InitEncoder}$ ,  $\text{EncodeDecision}$ ,  $\text{EncodeBypass}$  y  $\text{EncodeTerminate}$ , los cuales corresponden, respectivamente a  $\text{InitDecoder}$ ,  $\text{DecodeDecision}$ ,  $\text{DecodeBypass}$ , y  $\text{DecodeTerminate}$ . El estado del mecanismo de decodificación aritmética se representa mediante el valor de la variable  $\text{codILow}$  que apunta al final más pequeño de un subintervalo y el valor de la variable  $\text{codIRange}$  que especifica la correspondiente gama de subintervalo.

#### 9.3.4.1 Proceso de inicialización del mecanismo de codificación aritmética (informativo)

Esta subcláusula no es parte integrante de esta Recomendación | Norma Internacional.

Se llama a este proceso antes de codificar el primer macrobloque de un sector, y después de codificar el  $\text{pcm\_alignment\_zero\_bit}$  y todos los datos  $\text{pcm\_byte}$  de un macrobloque del tipo  $\text{I\_PCM}$ .

Este proceso genera como resultado los valores  $\text{codILow}$ ,  $\text{codIRange}$ ,  $\text{firstBitFlag}$ ,  $\text{bitsOutstanding}$ , y  $\text{symCnt}$  del mecanismo de codificación aritmética.

En el proceso de inicialización del codificador, codILow se pone a 0 y codIRange se pone a 0x01FE. Además, firstBitFlag se pone a 1 y los contadores bitsOutstanding y symCnt se ponen a 0.

NOTA – La precisión mínima de registro necesaria para codILow es de 10 bits y para CodIRange es de 9 bits. La precisión necesaria para los contadores bitsOutstanding y symCnt debe ser suficientemente grande para que no se produzca desbordamiento en los correspondientes registros. Cuando MaxBinCountInSlice indica el número total máximo de decisiones binarias para codificar un sector, la precisión mínima del registro necesaria para las variables bitsOutstanding y symCnt viene dado por  $\text{Ceil}(\text{Log}_2(\text{MaxBinCountInSlice} + 1))$ .

#### **9.3.4.2 Proceso de codificación de una decisión binaria (informativo)**

Esta subcláusula no es parte integrante de esta Recomendación | Norma Internacional.

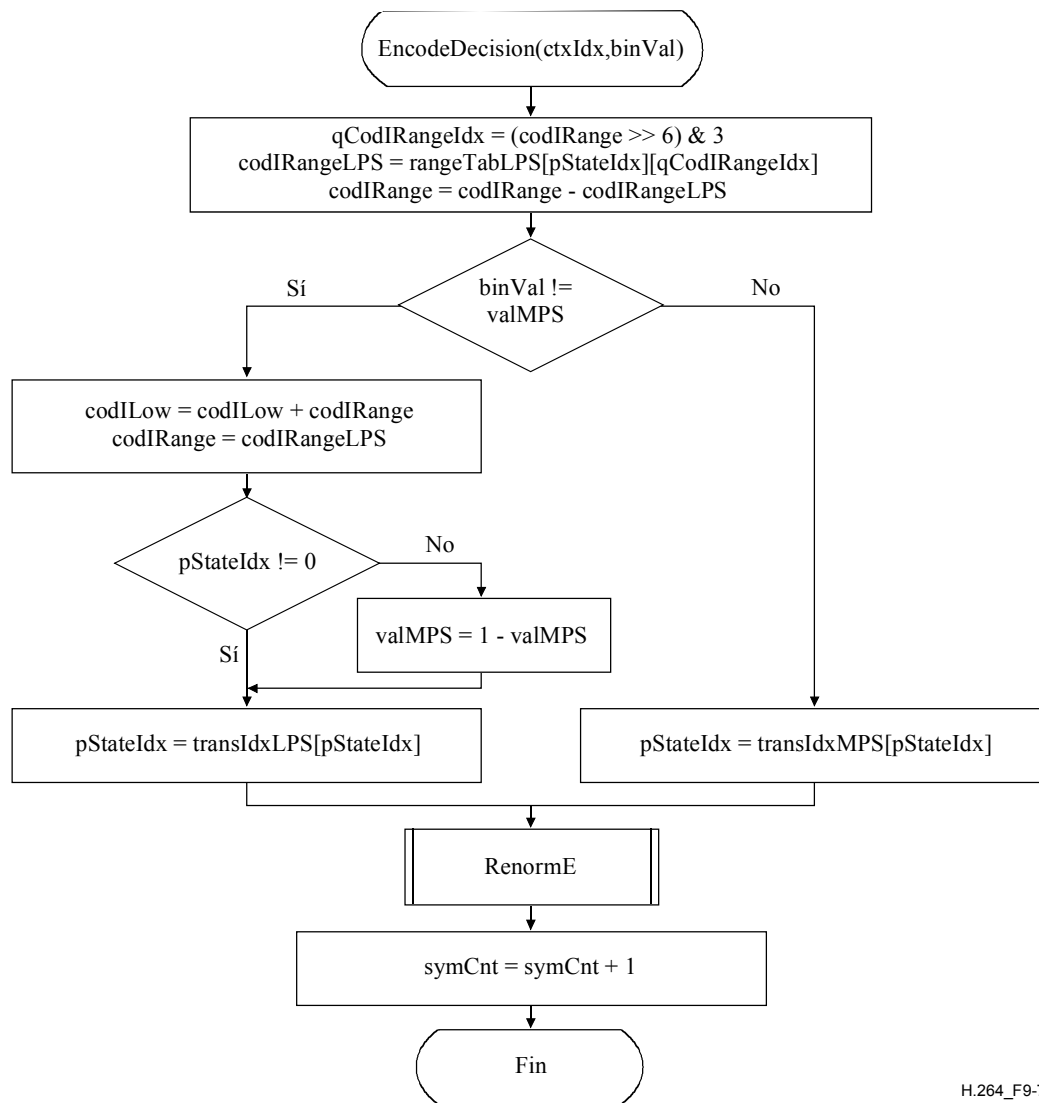
Este proceso acepta como argumentos el índice de contexto ctxIdx, el valor de binVal que se va a codificar y las variables codIRange, codILow y symCnt.

Este proceso genera como resultado las variables codIRange, codILow, y symCnt.

La figura 9-7 muestra el diagrama de flujo para codificar una sola decisión. En primer lugar, se calcula la variable codIRangeLPS como se describe a continuación.

Dado el valor actual de codIRange, codIRange se mapea en el índice qCodIRangeIdx de un valor cuantificado de codIRange mediante la ecuación (9-17). Se utilizan los valores de qCodIRangeIdx y de pStateIdx correspondiente a ctxIdx para determinar el valor de la variable rangeTabLPS como se describe en el cuadro 9-33, y este valor se asigna a codIRangeLPS. Se asigna a codIRange el valor  $\text{codIRange} - \text{codIRangeLPS}$ .

En el segundo paso, el valor de binVal se compara con el valMPS correspondiente a ctxIdx. Cuando binVal es diferente de valMPS, se suma codIRange a codILow y codIRange se pone igual al valor de codIRangeLPS. Dada la decisión codificada, la transición de estado se realiza como se describe en la subcláusula 9.3.3.2.1.1. En función del valor actual de codIRange, se aplica la renormalización como se describe en la subcláusula 9.3.4.3. Por último, la variable symCnt se incrementa en 1.



H.264\_F9-7

**Figura 9-7 – Diagrama de flujo de la codificación de una decisión**

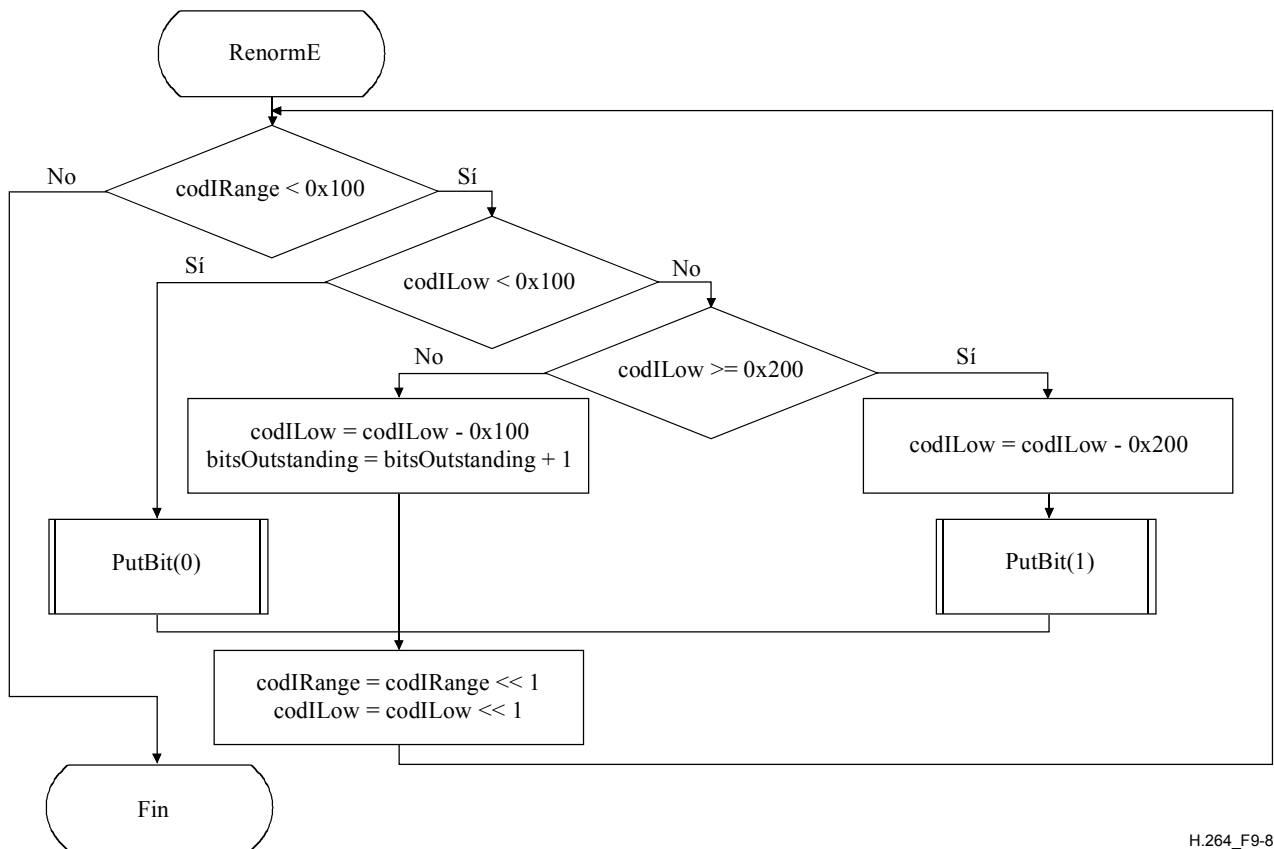
### 9.3.4.3 Proceso de renormalización en el mecanismo de codificación aritmética (informativo)

Esta subcláusula no forma parte integral de esta Recomendación | Norma Internacional.

Este proceso acepta como argumentos las variables `codIRange`, `codILow`, `firstBitFlag`, y `bitsOutstanding`.

Como resultado de aplicar este proceso se registran bits (ninguno o varios) en la RBSP y actualiza las variables `codIRange`, `codILow`, `firstBitFlag`, y `bitsOutstanding`.

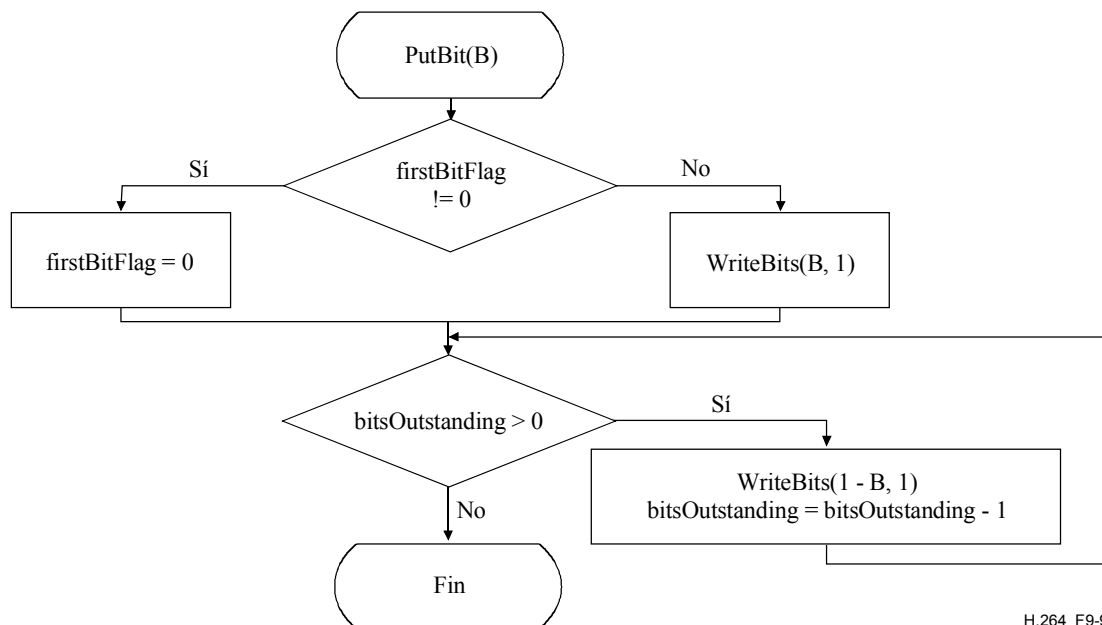
En la figura 9-8 se ilustra el proceso de renormalización.



H.264\_F9-8

**Figura 9-8 – Diagrama de flujo de la renormalización en el codificador**

El procedimiento PutBit() descrito en la figura 9-9 proporciona el control de acarreo. Este procedimiento utiliza la función WriteBits( B, N ) que escribe N bits de valor B en el tren de bits y avanza el puntero al tren de bits N posiciones de bit. Esta función supone la existencia de puntero al tren de bits que indica la posición en la que el proceso de codificación ha de escribir el siguiente bit en el tren de bits.



H.264\_F9-9

**Figura 9-9 – Diagrama de flujo de PutBit(B)**

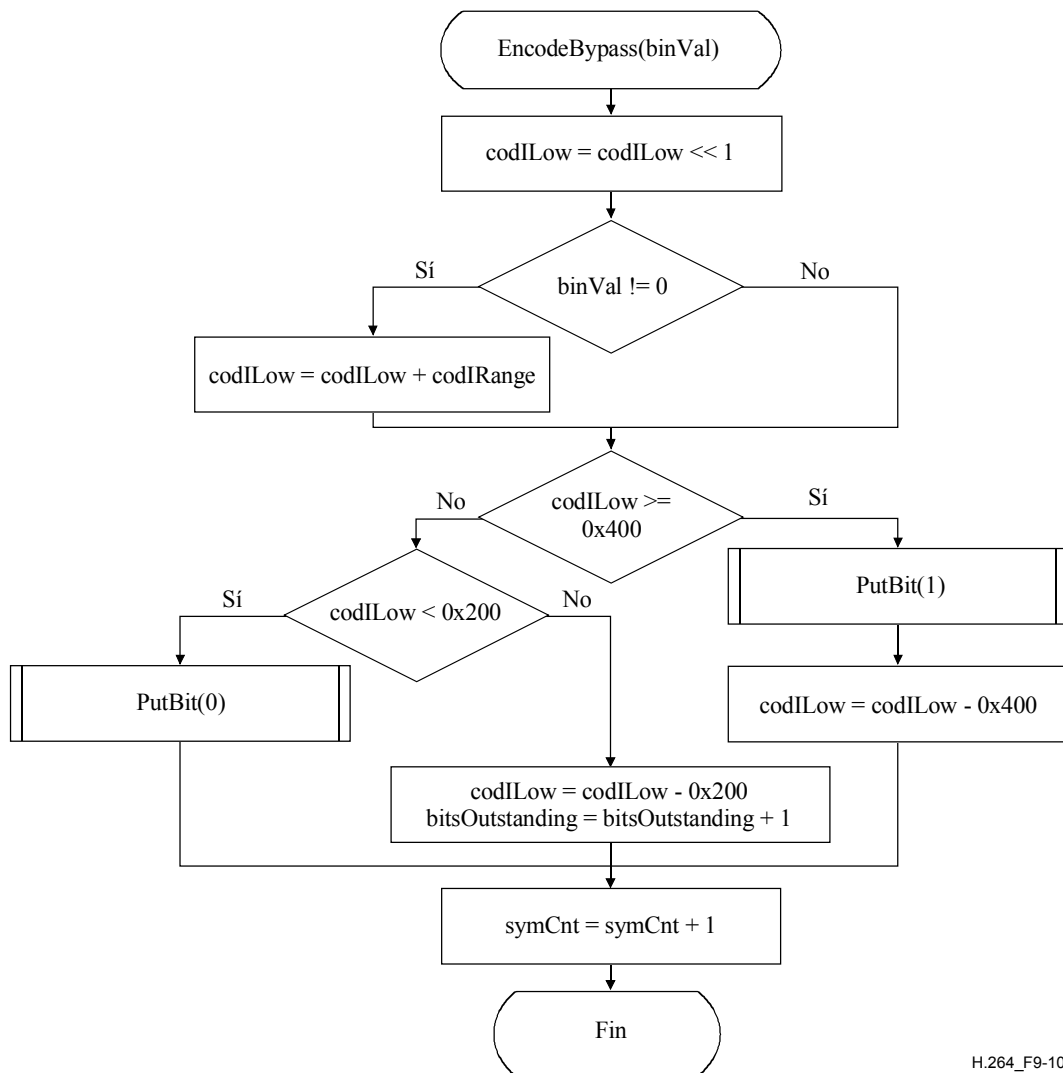
### 9.3.4.4 Proceso de codificación de derivación para decisiones binarias (informativo)

Esta subcláusula no forma parte integral de esta Recomendación | Norma Internacional.

Este proceso acepta como argumento las variables binVal, codILow, codIRange, bitsOutstanding, y symCnt.

Como resultado de aplicar este proceso se registra un bit en la RBSP y se actualizan las variables codILow, bitsOutstanding, y symCnt.

Este proceso de codificación se aplica a todas las decisiones binarias cuyo bypassFlag es igual a 1. En la especificación de este proceso se incluye la renormalización, como muestra la figura 9-10



H.264\_F9-10

Figura 9-10 – Diagrama de flujo de la codificación de derivación

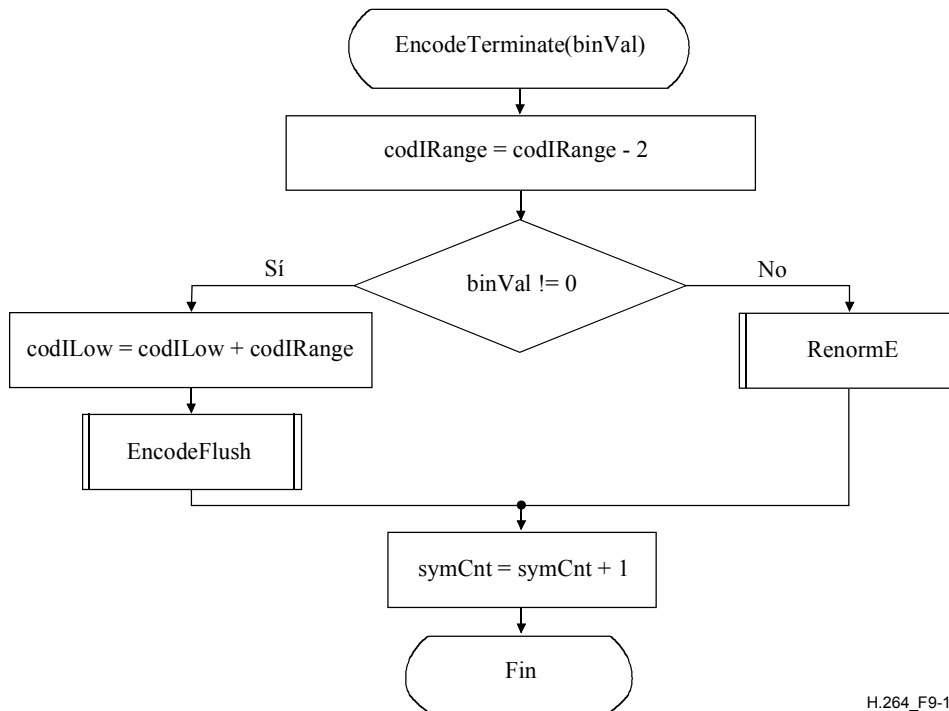
### 9.3.4.5 Proceso de codificación de una decisión binaria antes de la terminación (informativo)

Esta subcláusula no es parte integrante de esta Recomendación | Norma Internacional.

Este proceso acepta como argumento las variables binVal, codIRange, codILow, bitsOutstanding, y symCnt.

Como resultado de aplicar este proceso se registran bits (ninguno o varios) en la RBSP y se actualizan las variables codILow, codIRange, bitsOutstanding, y symCnt.

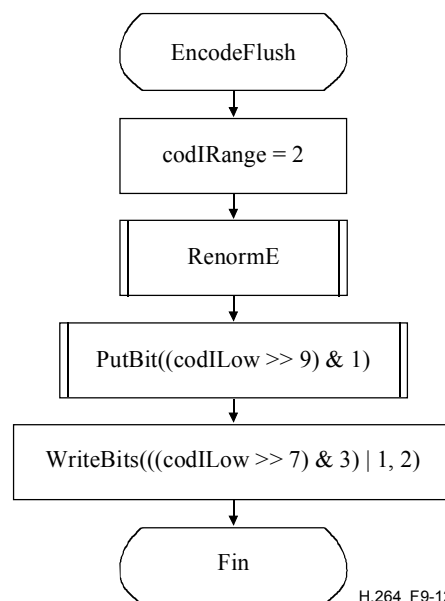
Esta rutina de codificación, que se muestra en la figura 9-11, se aplica a la codificación de end\_of\_slice\_flag y de bin que indica el I\_PCM mb\_type, ambos asociados con ctxIdx igual a 276.



H.264\_F9-11

**Figura 9-11 – Diagrama de flujo de la codificación de una decisión antes de la terminación**

Cuando el valor de binVal que se va a codificar es igual a 1, se termina la codificación CABAC y se aplica el procedimiento de evacuación que se muestra en la figura 9-12. En este procedimiento de evacuación el último bit escrito por WriteBits( B, N ) es igual a 1. Al codificar end\_of\_slice\_flag, este último bit se interpreta como el rbsp\_stop\_one\_bit.



H.264\_F9-12

**Figura 9-12 – Diagrama de flujo de la evacuación en la terminación**

### 9.3.4.6 Proceso de relleno de byte (informativo)

Esta subcláusula no es parte integrante de esta Recomendación | Norma Internacional.

Se llama este proceso después de codificar el último macrobloque del último sector de una imagen y después de la encapsulación.

Este proceso acepta como argumentos el número de bytes NumBytesInVclNALunits de todas las unidades NAL VCL de una imagen, el número de macrobloques PicSizeInMbs de la imagen, el número de símbolos binarios BinCountsInNALunits que resulta de codificar el contenido de todas las unidades NAL VCL de la imagen.

Como resultado de aplicar este proceso se añaden bytes (ninguno o varios) a la unidad NAL.

Sea la variable  $k$  definida por  $\text{Ceil}(\text{Ceil}(3 * \text{BinCountsInNALunits} - 3 * 96 * \text{PicSizeInMbs}) / 32) - \text{NumBytesInVclNALunits} / 3$ . En función de la variable  $k$  se aplica lo siguiente:

- Si  $k$  es menor o igual a 0, no se añade cabac\_zero\_word a la unidad NAL.
- De lo contrario ( $k$  es mayor que 0), se añade  $k$  veces la secuencia de tres bytes 0x000003 a la unidad NAL tras la encapsulación, donde los dos primeros bytes 0x0000 representan una cabac\_zero\_word y el tercer byte 0x03 representa un emulation\_prevention\_three\_byte.

## Anexo A

### Perfiles y niveles

(Este anexo es parte integrante de esta Recomendación | Norma Internacional)

Los perfiles y los niveles especifican restricciones aplicables a los trenes de bits y por consiguiente limitan las capacidades necesarias para decodificar los trenes de bits. Los perfiles y los niveles pueden utilizarse también para señalar puntos de interfuncionamiento entre aplicaciones particulares de los decodificadores.

NOTA – Esta Recomendación | Norma Internacional no incluye "opciones" que puedan seleccionarse individualmente en el decodificador, dado que esto dificultaría aún más el interfuncionamiento.

Cada perfil especifica un subconjunto de características y límites de los algoritmos que deben soportar todos los decodificadores conformes con ese perfil.

NOTA – No es obligatorio que los codificadores hagan uso de algún subconjunto particular de las prestaciones soportadas por un perfil.

Cada nivel especifica un conjunto de límites de los valores que pueden tomar los elementos sintácticos de esta Recomendación | Norma Internacional. Si bien todos los perfiles utilizan el mismo conjunto de definiciones de nivel, cada implementación puede soportar un nivel distinto por cada perfil soportado. Por lo general, para cualquier perfil determinado, los niveles corresponden a la carga computacional y a la capacidad de memoria del decodificador.

#### A.1 Requisitos relativos a la capacidad del decodificador de vídeo

Las capacidades de los decodificadores de vídeo conformes a esta Recomendación | Norma Internacional se especifican en términos de la aptitud para decodificar trenes de vídeo conformes a las restricciones de perfiles y niveles especificados en este anexo. Para cada perfil, se deberá expresar también el nivel soportado correspondiente a ese perfil.



En este anexo se especifican valores concretos de los elementos de sintaxis `profile_idc` y `level_idc`. El resto de los valores de `profile_idc` y `level_idc` se reservan para que el UIT-T | ISO/CEI pueda utilizarlos en el futuro.

NOTA – Cuando un valor reservado de `profile_idc` o `level_idc` está entre los valores especificados en esta Recomendación | Norma Internacional los decodificadores no deberían inferir que esto indica capacidades intermedias entre los perfiles o niveles especificados, ya que no existen restricciones sobre cómo utilizará el UIT-T | ISO/CEI esos valores reservados en el futuro.

## **A.2 Perfiles**

### **A.2.1 Perfil básico**

Los trenes de bits conformes al perfil básico deberán cumplir las siguientes restricciones:

- Sólo contendrán sectores de tipos I y P.
- Los trenes de unidades NAL no contendrán valores de `nal_unit_type` en la gama 2 a 4, inclusive.
- Los conjuntos de parámetros secuencia tendrán `frame_mbs_only_flag` igual a 1.
- Los conjuntos de parámetros imagen tendrán `weighted_pred_flag` y `weighted_bipred_idc` igual a 0 (ambos).
- Los conjuntos de parámetros imagen tendrán `entropy_coding_mode_flag` igual a 0.
- Los conjuntos de parámetros imagen tendrán `num_slice_groups_minus1` en la gama 0 a 7, inclusive.
- Se cumplirán las restricciones de nivel especificadas en la subcláusula A.3 para el perfil básico.

La conformidad de un tren de bits con el perfil básico se indica mediante `profile_idc` igual a 66.

Los decodificadores que son conformes con el perfil básico a un determinado nivel deberán ser capaces de decodificar todos los trenes de bits en los cuales `profile_idc` sea igual a 66 o `constraint_set0_flag` sea igual a 1 y en los que `level_idc` represente un nivel menor o igual que el nivel especificado.

### **A.2.2 Perfil principal**

Los trenes de bits conformes al perfil principal deberán cumplir las siguientes restricciones:

- Sólo contendrán sectores de tipos I, P y B.
- Los trenes de unidades NAL no contendrán valores de `nal_unit_type` en la gama 2 a 4, inclusive.
- No se permite el orden de sectores arbitrario.
- Los conjuntos de parámetros imagen tendrán `num_slice_groups_minus1` igual a 0, únicamente.
- Los conjuntos de parámetros imagen tendrán `redundant_pic_cnt_present_flag` igual a 0, únicamente.
- Se cumplirán las restricciones de nivel especificadas subcláusula A.3 para el perfil principal.

La conformidad con el tren de bits al perfil principal se indica mediante `profile_idc` igual a 77.

Los decodificadores conformes con el perfil principal a un determinado nivel deberán ser capaces de decodificar todos los trenes de bits en los que `profile_idc` sea igual a 77 o `constraint_set1_flag` sea igual a 1, y en los que `level_idc` represente un nivel menor o igual que el nivel especificado.

### A.2.3 Perfil extendido

Los trenes de bits conformes al perfil extendido cumplirán las siguientes restricciones:

- Los conjuntos de parámetros secuencia tendrán `direct_8x8_inference_flag` igual a 1.
- Los conjuntos de parámetros imagen tendrán `entropy_coding_mode_flag` igual a 0.
- Los conjuntos de parámetros imagen tendrán `num_slice_groups_minus1` en la gama 0 a 7, inclusive.
- Se cumplirán las restricciones de nivel especificadas en la subcláusula A.3 para el perfil extendido.

La conformidad de un tren de bits con el perfil extendido se indica mediante `profile_idc` igual a 88.

Los decodificadores conformes con el perfil extendido a un determinado nivel deberán ser capaces de decodificar todos los trenes de bits en los que `profile_idc` sea igual a 88 o `constraint_set2_flag` sea igual a 1 y en los que `level_idc` represente un nivel menor o igual que el nivel especificado.

Los decodificadores conformes con el perfil extendido a un determinado nivel también deberán ser capaces de decodificar todos los trenes de bits en los que `profile_idc` sea igual a 66 o `constraint_set0_flag` sea igual a 1, y en los que `level_idc` represente un nivel menor o igual que el nivel especificado.

### A.3 Niveles

Para expresar las limitaciones en este anexo se especifica lo siguiente.

- Sea la unidad de acceso  $n$  la  $n$ -ésima unidad de acceso en orden de decodificación, siendo la primera la unidad de acceso 0.
- Sea la imagen  $n$  la imagen codificada primaria o la correspondiente imagen decodificada de la unidad de acceso  $n$ .

#### A.3.1 Límites de nivel independientes del perfil

Sea  $fR$  una variable definida del modo siguiente.

- Si la imagen  $n$  es un cuadro,  $fR$  se pone a  $1 \div 172$ .
- De lo contrario (la imagen  $n$  es un campo),  $fR$  se pone a  $1 \div (172 * 2)$ .

Los trenes de bits conformes con algún perfil a un nivel especificado cumplirán las siguientes restricciones:

- a) Para el instante de extracción nominal de la unidad de acceso  $n$  (con  $n > 0$ ) de la memoria intermedia de imágenes codificadas (CPB, *coded picture buffer*), según se especifica en la subcláusula C.1.2, se cumple que  $t_{r,n}(n) - t_r(n-1)$  es mayor o igual que  $\text{Max}(\text{PicSizeInMbs} \div \text{MaxMBPS}, fR)$ , donde  $\text{MaxMBPS}$  es el valor indicado en el cuadro A-1 para la imagen  $n$ , y  $\text{PicSizeInMbs}$  es el número de macrobloques de la imagen  $n$ .
- b) Para la diferencia entre instantes de salida consecutivos de imágenes de la CPB, según se define en la subcláusula C.2.2, se cumple que  $\Delta t_{o,dpb}(n) \geq \text{Max}(\text{PicSizeInMbs} \div \text{MaxMBPS}, fR)$ , donde  $\text{MaxMBPS}$  es el valor indicado en el cuadro A-1 para la imagen  $n$ , y  $\text{PicSizeInMbs}$  es el número de macrobloques de la imagen  $n$ , siempre que la imagen  $n$  sea una imagen de salida y no sea la última imagen del tren de bits de salida.
- c) La suma de las variables  $\text{NumBytesInNALunit}$  de la unidad de acceso 0 es menor o igual a  $256 * \text{ChromaFormatFactor} * (\text{PicSizeInMbs} + \text{MaxMBPS} * (t_r(0) - t_{r,n}(0))) \div \text{MinCR}$ , donde  $\text{MaxMBPS}$  y  $\text{MinCR}$  son los valores indicados en el

- cuadro A-1 para la imagen 0 y PicSizeInMbs es el número de macrobloques de la imagen 0.
- d) La suma de las variables NumBytesInNALunit de la unidad de acceso n (con  $n > 0$ ) es menor o igual que  $256 * \text{ChromaFormatFactor} * \text{MaxMBPS} * (t_r(n) - t_r(n-1)) \div \text{MinCR}$ , donde MaxMBPS y MinCR son los valores indicados en el cuadro A-1 para la imagen n.
  - e)  $\text{PicWidthInMbs} * \text{FrameHeightInMbs} \leq \text{MaxFS}$ , donde MaxFS se indica en el cuadro A-1.
  - f)  $\text{PicWidthInMbs} \leq \text{Sqrt}(\text{MaxFS} * 8)$ .
  - g)  $\text{FrameHeightInMbs} \leq \text{Sqrt}(\text{MaxFS} * 8)$ .
  - h)  $\text{max\_dec\_frame\_buffering} \leq \text{MaxDpbSize}$ , donde MaxDpbSize es igual a  $\text{Min}(1024 * \text{MaxDPB} / (\text{PicWidthInMbs} * \text{FrameHeightInMbs} * 256 * \text{ChromaFormatFactor}), 16)$  y MaxDPB se indica en el cuadro A-1 en unidades de 1024 bytes. max\_dec\_frame\_buffering se denomina también tamaño de la DPB.
  - i) Para los parámetros HRD VCL,  $\text{BitRate}[\text{SchedSelIdx}] \leq 1000 * \text{MaxBR}$  y  $\text{CpbSize}[\text{SchedSelIdx}] \leq 1000 * \text{MaxCPB}$  al menos para un valor de SchedSelIdx, donde BitRate[SchedSelIdx] viene dada por la ecuación E-13 y CpbSize[SchedSelIdx] por la ecuación E-14 cuando vcl\_hrd\_parameters\_present\_flag es igual a 1. MaxBR y MaxCPB se indican en el cuadro A-1 en unidades de 1000 bits/s y 1000 bits, respectivamente. El tren de bits cumplirá estas condiciones al menos para un valor de SchedSelIdx en la gama 0 a cpb\_cnt\_minus1, inclusive. CpbSize[SchedSelIdx] se denomina también tamaño de la CPB.
  - j) Para los parámetros HRD NAL,  $\text{BitRate}[\text{SchedSelIdx}] \leq 1200 * \text{MaxBR}$  y  $\text{CpbSize}[\text{SchedSelIdx}] \leq 1200 * \text{MaxCPB}$  al menos para un valor de SchedSelIdx, donde BitRate[SchedSelIdx] viene dada por la ecuación E-13 y CpbSize[SchedSelIdx] por la ecuación E-14 cuando nal\_hrd\_parameters\_present\_flag es igual a 1. MaxBR y MaxCPB se indican en el cuadro A-1 en unidades de 1200 bits/s y 1200 bits, respectivamente. El tren de bits cumplirá estas condiciones al menos para un valor de SchedSelIdx en la gama 0 a cpb\_cnt\_minus1.
  - k) La gama de los componentes de vectores de movimiento vertical no debe ser superior a MaxVmvR en unidades de muestra cuadro luma, donde MaxVmvR se indica en el cuadro A-1.
  - l) La gama de vectores de movimiento horizontal no debe ser superior a la gama de -2048 a 2047,75, inclusive, en unidades de muestras luma.
  - m) El número de vectores de movimiento por cada dos macrobloques consecutivos en orden de decodificación (también se aplica al número total entre el último macrobloque de un sector y el primer macrobloque del siguiente sector en el orden de decodificación) no debe ser superior a MaxMvsPer2Mb, donde MaxMvsPer2Mb se indica en el cuadro A-1. La cantidad de vectores de movimiento en cada macrobloque es igual al valor de la variable MvCnt después de que finalizan el proceso de predicción intra o inter para el macrobloque.
  - n) El número de bits de los datos de macroblock\_layer() para cualquier macrobloque no debe ser mayor que  $128 + 2048 * \text{ChromaFormatFactor}$ . En función de entropy\_coding\_mode\_flag, los bits de los datos de macroblock\_layer() se cuentan del modo siguiente:
    - si entropy\_coding\_mode\_flag es igual a 0, el número de bits de los datos macroblock\_layer() será igual al número de bits en la estructura de sintaxis de macroblock\_layer() de un macrobloque;

- de lo contrario (entropy\_coding\_mode\_flag es igual a 1), el número de bits de los datos de macroblock\_layer( ) de un macrobloque será igual al número de veces que se invoca read\_bits( 1 ), de acuerdo con las subcláusulas 9.3.3.2.2 y 9.3.3.2.3, cuando se analiza la macroblock\_layer( ) correspondiente al macrobloque.

En el cuadro A-1 se indican los límites de cada nivel. Las casillas marcadas con "-" indican que no hay un límite aplicable al elemento correspondiente.

La conformidad con un nivel particular se especificará mediante el elemento de sintaxis level\_idc cuyo valor será igual a 10 veces el número del nivel indicado en el cuadro A-1.

**Cuadro A-1 – Límites de nivel**

Número de nivel	Velocidad máxima de procesamiento de macrobloque MaxMBPS (MB/s)	Tamaño máximo de cuadro MaxFS (MBs)	Tamaño máximo de la memoria intermedia de imágenes decodificadas MaxDPB (1024 bytes)	Velocidad binaria máxima de vídeo MaxBR (1000 bits/s o 1200 bits/s)	Tamaño máximo de la CPB MaxCPB (1000 bits o 1200 bits)	Gama de componentes de vector de movimiento (MV) vertical MaxVmvR (muestras cuadro luma)	Relación de compresión mínima MinCR	Número máximo de vectores de movimiento por cada dos macrobloques (MB) consecutivos MaxMvsPer2Mb
1	1 485	99	148,5	64	175	[-64,+63,75]	2	–
1.1	3 000	396	337,5	192	500	[-128,+127,75]	2	–
1.2	6 000	396	891,0	384	1 000	[-128,+127,75]	2	–
1.3	11 880	396	891,0	768	2 000	[-128,+127,75]	2	–
2	11 880	396	891,0	2 000	2 000	[-128,+127,75]	2	–
2.1	19 800	792	1 782,0	4 000	4 000	[-256,+255,75]	2	–
2.2	20 250	1 620	3 037,5	4 000	4 000	[-256,+255,75]	2	–
3	40 500	1 620	3 037,5	10 000	10 000	[-256,+255,75]	2	32
3.1	108 000	3 600	6 750,0	14 000	14 000	[-512,+511,75]	4	16
3.2	216 000	5 120	7 680,0	20 000	20 000	[-512,+511,75]	4	16
4	245 760	8 192	12 288,0	20 000	25 000	[-512,+511,75]	4	16
4.1	245 760	8 192	12 288,0	50 000	62 500	[-512,+511,75]	2	16
4.2	491 520	8 192	12 288,0	50 000	62 500	[-512,+511,75]	2	16
5	589 824	22 080	41 310,0	135 000	135 000	[-512,+511,75]	2	16
5.1	983 040	36 864	69 120,0	240 000	240 000	[-512,+511,75]	2	16

Los niveles cuyo número de nivel en el cuadro A-1 no es un valor entero se denominan "niveles intermedios".

NOTA – Si bien no hay jerarquías entre los niveles, es posible que algunas aplicaciones utilicen únicamente los niveles enumerados con valores enteros.

En la subcláusula A.3.3 informativa se ilustra el efecto que tienen estos límites en las velocidades de cuadro para varios ejemplos de formatos de imagen.

### A.3.2 Límites de nivel específico del perfil

- En los trenes de bits conformes con el perfil principal, el instante de extracción de la unidad de acceso 0 será tal que el número de sectores en la imagen 0 sea menor o igual que  $(PicSizeInMbs + MaxMBPS * (t_r(0) - t_{r,n}(0))) \div SliceRate$ , siendo SliceRate el valor indicado en el cuadro A-3 para la imagen 0.
- En los trenes de bits conformes al perfil principal, la diferencia entre el instante de extracción consecutivo de las unidades de acceso n y n – 1 (con n > 0) será tal que el número de sectores en la imagen n sea menor o igual que  $MaxMBPS *$

- $(t_r(n) - t_r(n-1)) \div \text{SliceRate}$ , siendo SliceRate el valor indicado en el cuadro A-3 para la imagen n.
- c) En los trenes de bits conformes al perfil principal, los conjuntos de parámetros secuencia tendrán `direct_8x8_inference_flag` igual a 1 para los niveles indicados en el cuadro A-3.
- NOTA – `direct_8x8_inference_flag` es irrelevante para el perfil básico, dado que no acepta sectores de tipo B (conforme a la subcláusula A.2.1), y `direct_8x8_inference_flag` es igual a 1 para todos los niveles del perfil extendido (conforme a la subcláusula A.2.3).
- d) En los trenes de bits conformes a los perfiles principal y extendido, los conjuntos de parámetros secuencia tendrán `frame_mbs_only_flag` igual a 1 en los niveles indicados en el cuadro A-3 para el perfil principal y en el cuadro A-4 para el perfil extendido.
- NOTA – `frame_mbs_only_flag` es igual a 1 para todos los niveles del perfil básico (conforme a la subcláusula A.2.1).
- e) En los trenes de bits conformes con los perfiles principal y extendido, el valor de `sub_mb_type` en los macrobloques B será distinto de `B_Bi_8x4`, `B_Bi_4x8` y `B_Bi_4x4` en los niveles cuyo `MinLumaBiPredSize` es igual a  $8 \times 8$  en el cuadro A-3 para el perfil principal y en el cuadro A-4 para el perfil extendido.
- f) En los trenes de bits conformes a los perfiles básico y extendido, se cumple  $(xInt_{max} - xInt_{min} + 6) * (yInt_{max} - yInt_{min} + 6) \leq \text{MaxSubMbRectSize}$  en los macrobloques codificados con el `mb_type` igual a `P_8x8`, `P_8x8ref0` o `B_8x8` todas las veces que se llama al proceso descrito en la subcláusula 8.4.2.2.1 que se utiliza para generar la matriz de muestras luma predichas correspondiente a una sola lista (lista 0 o lista 1) para cada submacrobloque  $8 \times 8$ , siendo  $\text{NumSubMbPart}(\text{sub\_mb\_type}) > 1$ , y el valor de `MaxSubMbRectSize` se indica en el cuadro A-2 para el perfil básico y en el cuadro A-4 para el perfil extendido, y siendo
- $xInt_{min}$  el valor mínimo de  $xInt_L$  entre todas las posibles muestras luma del submacrobloque;
  - $xInt_{max}$  el máximo valor de  $xInt_L$  entre todas las posibles muestras luma del submacrobloque;
  - $yInt_{min}$  el valor mínimo de  $yInt_L$  entre todas las posibles muestras luma del submacrobloque;
  - $yInt_{max}$  el máximo valor de  $yInt_L$  entre todas las posibles muestras luma del submacrobloque.

### A.3.2.1 Límites del perfil básico

En el cuadro A-2 se indican los límites correspondientes a cada nivel de los trenes de bits conformes con el perfil básico. En este cuadro el símbolo "-" indica que no se aplica límite al nivel correspondiente.

**Cuadro A-2 – Límites de nivel del perfil básico**

Level number	MaxSubMbRectSize
1	576
1.1	576
1.2	576
1.3	576
2	576
2.1	576
2.2	576
3	576
3.1	-
3.2	-
4	-
4.1	-
4.2	-
5	-
5.1	-

**A.3.2.2 Límites del perfil principal**

En el cuadro A-3 se indican los límites correspondientes a cada nivel de los trenes de bits conformes con el perfil principal. En este cuadro el símbolo "-" indica que no se aplica límite al nivel correspondiente.

**Cuadro A-3 – Límites de nivel del perfil principal**

Número de nivel	SliceRate	MinLumaBiPredSize	direct_8x8_inference_flag	frame_mbs_only_flag
1	-	-	-	1
1.1	-	-	-	1
1.2	-	-	-	1
1.3	-	-	-	1
2	-	-	-	1
2.1	-	-	-	-
2.2	-	-	-	-
3	22	-	1	-
3.1	60	8x8	1	-
3.2	60	8x8	1	-
4	60	8x8	1	-
4.1	24	8x8	1	-
4.2	24	8x8	1	1
5	24	8x8	1	1
5.1	24	8x8	1	1

### A.3.2.3 Límites del perfil extendido

En el cuadro A-4 se indican los límites correspondientes a cada nivel de los trenes de bits conformes con el perfil extendido. En este cuadro el símbolo "-" indica que no se aplica límite al nivel correspondiente.

**Cuadro A-4 – Límites de nivel del perfil extendido**

Número de nivel	MaxSubMbRectSize	MinLumaBiPredSize	frame_mbs_only_flag
1	576	–	1
1.1	576	–	1
1.2	576	–	1
1.3	576	–	1
2	576	–	1
2.1	576	–	–
2.2	576	–	–
3	576	–	–
3.1	–	8x8	–
3.2	–	8x8	–
4	–	8x8	–
4.1	–	8x8	–
4.2	–	8x8	1
5	–	8x8	1
5.1	–	8x8	1

### A.3.3 Efecto de los límites de nivel en la velocidad de cuadro (informativo)

Esta subcláusula no es parte integrante de esta Recomendación | Norma Internacional.

**Cuadro A-5 – Velocidades máximas de cuadro (cuadros por segundo)  
para algunos ejemplos de tamaños de cuadros**

Número de nivel:					1	1.1	1.2	1.3	2	2.1	2.2
Tamaño máximo de cuadro (macrobloques):					99	396	396	396	396	792	1 620
Macrobloques/s max:					1 485	3 000	6 000	11 880	11 880	19 800	20 250
Tamaño máximo de cuadro (muestras):					25 344	101 376	101 376	101 376	101 376	202 752	414 720
Muestras/s max:					380 160	768 000	1 536 000	3 041 280	3 041 280	5 068 800	5 184 000
Formato	Anchura luma	Altura luma	Total MB	Muestras luma							
SQCIF	128	96	48	12 288	30,9	62,5	125,0	172,0	172,0	172,0	172,0
QCIF	176	144	99	25 344	15,0	30,3	60,6	120,0	120,0	172,0	172,0
QVGA	320	240	300	76 800	–	10,0	20,0	39,6	39,6	66,0	67,5
525 SIF	352	240	330	84 480	–	9,1	18,2	36,0	36,0	60,0	61,4
CIF	352	288	396	101 376	–	7,6	15,2	30,0	30,0	50,0	51,1
525 HHR	352	480	660	168 960	–	–	–	–	–	30,0	30,7
625 HHR	352	576	792	202 752	–	–	–	–	–	25,0	25,6
VGA	640	480	1 200	307 200	–	–	–	–	–	–	16,9
525 4SIF	704	480	1 320	337 920	–	–	–	–	–	–	15,3
525 SD	720	480	1 350	345 600	–	–	–	–	–	–	15,0
4CIF	704	576	1 584	405 504	–	–	–	–	–	–	12,8
625 SD	720	576	1 620	414 720	–	–	–	–	–	–	12,5
SVGA	800	600	1 900	486 400	–	–	–	–	–	–	–
XGA	1024	768	3 072	786 432	–	–	–	–	–	–	–
720p HD	1280	720	3 600	921 600	–	–	–	–	–	–	–
4VGA	1280	960	4 800	1 228 800	–	–	–	–	–	–	–
SXGA	1280	1024	5 120	1 310 720	–	–	–	–	–	–	–
525 16SIF	1408	960	5 280	1 351 680	–	–	–	–	–	–	–
16CIF	1408	1152	6 336	1 622 016	–	–	–	–	–	–	–
4SVGA	1600	1200	7 500	1 920 000	–	–	–	–	–	–	–
1080 HD	1920	1088	8 160	2 088 960	–	–	–	–	–	–	–
2Kx1K	2048	1024	8 192	2 097 152	–	–	–	–	–	–	–
4XGA	2048	1536	12 288	3 145 728	–	–	–	–	–	–	–
16VGA	2560	1920	19 200	4 915 200	–	–	–	–	–	–	–
3616x1536 (2.35:1)	3616	1536	21 696	5 554 176	–	–	–	–	–	–	–
3672x1536 (2.39:1)	3680	1536	22 080	5 652 480	–	–	–	–	–	–	–
4Kx2K	4096	2048	32 768	8 388 608	–	–	–	–	–	–	–
4096x2304 (16:9)	4096	2304	36 864	9 437 184	–	–	–	–	–	–	–



**Cuadro A-5 – (continuación) – Velocidades máximas de cuadro (cuadros por segundo)  
para algunos ejemplos de tamaños de cuadros**

Número de nivel:					3	3.1	3.2	4	4.1	4.2
Tamaño máximo de cuadro (macrobloques):					1 620	3 600	5 120	8 192	8 192	8 192
Macrobloques/s max:					40 500	108 000	216 000	245 760	245 760	589 824
Tamaño máximo de cuadro (muestras):					414 720	921 600	1 310 720	2 097 152	2 097 152	2 097 152
Muestras/s max:					10 368 000	27 648 000	55 296 000	62 914 560	62 914 560	125 829 120
Formato	Anchura luma	Altura luma	Total MB	Muestras luma						
SQCIF	128	96	48	12 288	172,0	172,0	172,0	172,0	172,0	172,0
QCIF	176	144	99	25 344	172,0	172,0	172,0	172,0	172,0	172,0
QVGA	320	240	300	76 800	135,0	172,0	172,0	172,0	172,0	172,0
525 SIF	352	240	330	84 480	122,7	172,0	172,0	172,0	172,0	172,0
CIF	352	288	396	101 376	102,3	172,0	172,0	172,0	172,0	172,0
525 HHR	352	480	660	168 960	61,4	163,6	172,0	172,0	172,0	172,0
625 HHR	352	576	792	202 752	51,1	136,4	172,0	172,0	172,0	172,0
VGA	640	480	1 200	307 200	33,8	90,0	172,0	172,0	172,0	172,0
525 4SIF	704	480	1 320	337 920	30,7	81,8	163,6	172,0	172,0	172,0
525 SD	720	480	1 350	345 600	30,0	80,0	160,0	172,0	172,0	172,0
4CIF	704	576	1 584	405 504	25,6	68,2	136,4	155,2	155,2	172,0
625 SD	720	576	1 620	414 720	25,0	66,7	133,3	151,7	151,7	172,0
SVGA	800	600	1 900	486 400	–	56,8	113,7	129,3	129,3	172,0
XGA	1024	768	3 072	786 432	–	35,2	70,3	80,0	80,0	160,0
720p HD	1280	720	3 600	921 600	–	30,0	60,0	68,3	68,3	136,5
4VGA	1280	960	4 800	1 228 800	–	–	45,0	51,2	51,2	102,4
SXGA	1280	1024	5 120	1 310 720	–	–	42,2	48,0	48,0	96,0
525 16SIF	1408	960	5 280	1 351 680	–	–	–	46,5	46,5	93,1
16CIF	1408	1152	6 336	1 622 016	–	–	–	38,8	38,8	77,6
4SVGA	1600	1200	7 500	1 920 000	–	–	–	32,8	32,8	65,5
1080 HD	1920	1088	8 160	2 088 960	–	–	–	30,1	30,1	60,2
2Kx1K	2048	1024	8 192	2 097 152	–	–	–	30,0	30,0	60,0
4XGA	2048	1536	12 288	3 145 728	–	–	–	–	–	–
16VGA	2560	1920	19 200	4 915 200	–	–	–	–	–	–
3616x1536 (2.35:1)	3616	1536	21 696	5 554 176	–	–	–	–	–	–
3672x1536 (2.39:1)	3680	1536	22 080	5 652 480	–	–	–	–	–	–
4Kx2K	4096	2048	32 768	8 388 608	–	–	–	–	–	–
4096x2304 (16:9)	4096	2304	36 864	9 437 184	–	–	–	–	–	–

**Cuadro A-5 – (fin) – Velocidades máximas de cuadro (cuadros por segundo)  
para algunos ejemplos de tamaños de cuadros**

Número de nivel:					5	5.1
Tamaño máximo de cuadro (macrobloques):					21 696	36 864
Macrobloques/s max:					589 824	983 040
Tamaño máximo de cuadro (muestras):					5 554 176	9 437 184
Muestras/s max:					150 994 944	251 658 240
<b>Formato</b>	<b>Anchura de muestra luma</b>	<b>Altura de muestra luma</b>	<b>Total MB</b>	<b>Muestras luma</b>		
SQCIF	128	96	48	12 288	172,0	172,0
QCIF	176	144	99	25 344	172,0	172,0
QVGA	320	240	300	76 800	172,0	172,0
525 SIF	352	240	330	84 480	172,0	172,0
CIF	352	288	396	101 376	172,0	172,0
525 HHR	352	480	660	168 960	172,0	172,0
625 HHR	352	576	792	202 752	172,0	172,0
VGA	640	480	1 200	307 200	172,0	172,0
525 4SIF	704	480	1 320	337 920	172,0	172,0
525 SD	720	480	1 350	345 600	172,0	172,0
4CIF	704	576	1 584	405 504	172,0	172,0
625 SD	720	576	1 620	414 720	172,0	172,0
SVGA	800	600	1 900	486 400	172,0	172,0
XGA	1024	768	3 072	786 432	172,0	172,0
720p HD	1280	720	3 600	921 600	163,8	172,0
4VGA	1280	960	4 800	1 228 800	122,9	172,0
SXGA	1280	1024	5 120	1 310 720	115,2	172,0
525 16SIF	1408	960	5 280	1 351 680	111,7	172,0
16CIF	1408	1152	6 336	1 622 016	93,1	155,2
4SVGA	1600	1200	7 500	1 920 000	78,6	131,1
1080 HD	1920	1088	8 160	2 088 960	72,3	120,5
2Kx1K	2048	1024	8 192	2 097 152	72,0	120,0
4XGA	2048	1536	12 288	3 145 728	48,0	80,0
16VGA	2560	1920	19 200	4 915 200	30,7	51,2
3616x1536 (2.35:1)	3616	1536	21 696	5 554 176	27,2	45,3
3672x1536 (2.39:1)	3680	1536	22 080	5 652 480	26,7	44,5
4Kx2K	4096	2048	32 768	8 388 608	–	30,0
4096x2304 (16:9)	4096	2304	36 864	9 437 184	–	26,7

Cabe observar lo siguiente:

- Esta Recomendación | Norma Internacional es una especificación en la que el tamaño de cuadro es variable. Los tamaños de cuadro indicados en el cuadro A-5 son únicamente ejemplos ilustrativos.
- En el cuadro A-5, "525" se refiere al uso normal en contextos que emplean 525 líneas de barrido analógico (de las cuales aproximadamente 480 abarcan la región visible de la imagen), y "625" se refiere a contextos que utilizan 625 líneas de barrido analógico (de las cuales aproximadamente 576 abarcan la región visible de la imagen).

- Las siguientes siglas son sinónimos XGA y XVGA; 4SVGA y UXGA; 16XGA y 4Kx3K; CIF y 625 SIF; 625 HHR, 2CIF, mitad 625 D-1, mitad 625 UIT-R BT.601; 525 SD, 525 D-1 y 525 UIT-R BT.601; 625 SD, 625 D-1 y 625 UIT-R BT.601.
- Las velocidades de cuadro indicadas son adecuadas para los modos de barrido progresivo y también para la codificación de vídeo entrelazado, pero en este último caso sólo cuando la altura de cuadro es divisible por 32.

## Anexo B

### Formato del tren de bytes

(Este anexo es parte integrante de esta Recomendación | Norma Internacional)

En este anexo se especifican la sintaxis y la semántica de un formato de tren de bytes que se puede utilizar en aplicaciones que distribuyen algunos o todos los trenes de unidades NAL en la forma de un tren ordenado de bytes o de bits, y dentro del cual es necesario identificar los límites de las unidades NAL con respecto a patrones de los datos, por ejemplo la Recomendación UIT-T H.222.0 | ISO/CEI 13818-1 o la Recomendación UIT-T H.320. En la distribución orientada a bits, el orden de los bits del formato del tren de bytes comienza con el bit más significativo (MSB) del primer byte hasta el bit menos significativo (LSB) del primer byte, seguido del MSB del segundo byte, etc.

El formato del tren de bytes consiste en una secuencia de estructuras sintácticas unidad NAL de tren de bytes. Cada estructura de este tipo contiene un prefijo código de inicio seguido por una estructura sintáctica `nal_unit(NumBytesInNALunit)`. Además puede contener (en ciertas circunstancias es obligatorio) un elemento sintáctico `zero_byte` adicional. También puede contener uno o más elementos sintácticos `trailing_zero_8bits` adicionales. Si se trata de la primera unidad NAL del tren de bytes en el tren de bits, puede también incluir uno o más elementos sintácticos adicionales `leading_zero_8bits`.

#### B.1 Sintaxis y semántica de unidades NAL de tren de bytes

##### B.1.1 Sintaxis de unidades NAL de tren de bytes

	C	Descriptor
<code>byte_stream_nal_unit( NumBytesInNALunit ) {</code>		
<code>  while( next_bits( 24 ) != 0x000001 &amp;&amp;         next_bits( 32 ) != 0x00000001 )</code>		
<code>    <b>leading_zero_8bits</b> /* igual a 0x00 */</code>		f(8)
<code>  if( next_bits( 24 ) != 0x000001 )</code>		
<code>    <b>zero_byte</b> /* igual a 0x00 */</code>		f(8)
<code>  if( more_data_in_byte_stream( ) ) {</code>		
<code>    <b>start_code_prefix_one_3bytes</b> /* igual a 0x000001 */</code>		f(24)
<code>    nal_unit( NumBytesInNALunit )</code>		
<code>  }</code>		
<code>  while( more_data_in_byte_stream( ) &amp;&amp;         next_bits( 24 ) != 0x000001 &amp;&amp;         next_bits( 32 ) != 0x00000001 )</code>		
<code>    <b>trailing_zero_8bits</b> /* igual a 0x00 */</code>		f(8)
<code>}</code>		

## B.1.2 Semántica de unidades NAL de tren de bytes

El orden de las unidades NAL de tren de bytes en el tren de bytes tiene que ser idéntico al orden de decodificación de las unidades NAL incluidas en las unidades NAL de tren de bytes (véase la subcláusula 7.4.1.2). El contenido de cada unidad NAL de tren de bytes se asocia con la misma unidad de acceso de la unidad NAL incluida en la unidad NAL de tren de bytes (véase la subcláusula 7.4.1.2.3).

**leading\_zero\_8bits** es un byte igual a 0x00.

NOTA – El elemento sintáctico **leading\_zero\_8bits** puede darse solamente en la primera unidad NAL del tren de bytes en el tren de bits, debido a que (como se muestra en el diagrama sintáctico de la subcláusula B.1.1) se considera que los bytes iguales a 0x00 que siguen a una estructura sintáctica unidad NAL y anteceden la secuencia de cuatro bytes 0x00000001 (la cual se interpreta como un **zero\_byte** seguido por un **start\_code\_prefix\_one\_3bytes**) son considerados elementos sintácticos **trailing\_zero\_8bits** que forman parte de la unidad NAL del tren de bytes anterior.

**zero\_byte** es un solo byte de valor igual a 0x00.

Cuando se cumple alguna de las siguientes condiciones, el elemento sintáctico **zero\_byte** estará presente.

- el **nal\_unit\_type** dentro de **nal\_unit()** es igual a 7 (conjunto de parámetros secuencia) u 8 (conjunto de parámetros imagen);
- la estructura sintáctica unidad NAL de tren de bytes contiene la primera unidad NAL de una unidad de acceso en el orden de decodificación, conforme a la subcláusula 7.4.1.2.3.

**start\_code\_prefix\_one\_3bytes** es una secuencia de valor constante de 3 bytes igual a 0x000001. Este elemento sintáctico se denomina prefijo código de inicio.

**trailing\_zero\_8bits** es un byte igual a 0x00.

## B.2 Proceso de decodificación de unidades NAL de tren de bytes

Este proceso acepta como argumento un tren ordenado de bytes que consiste en una secuencia de estructuras sintácticas unidad NAL de tren de bytes.

Este proceso genera como resultado una secuencia de estructuras sintácticas unidad NAL.

Al comienzo del proceso de decodificación, el decodificador inicializa su posición en el tren de bytes al principio de éste. Luego extrae y descarta los elementos sintácticos **leading\_zero\_8bits** (si están presentes), avanzando byte por byte la posición actual en el tren de bytes, hasta que la posición actual en el tren de bytes sea tal que los siguientes cuatro bytes del tren de bits conformen la secuencia de cuatro bytes 0x00000001.

A continuación, el decodificador realiza reiteradamente el siguiente proceso por pasos para extraer y decodificar cada una de las estructuras sintácticas unidad NAL del tren de bytes, hasta que se llega al final del tren de bytes (no se especifica cómo se determina el final del tren de bytes) y la última unidad NAL del tren de bytes ha sido decodificada:

1. Una vez que los siguientes cuatro bytes del tren de bits conformen la secuencia de cuatro bytes 0x00000001, se extrae y descarta el siguiente byte del tren de bytes (que es un elemento sintáctico **zero\_byte**) y la posición actual en el tren de bytes se fija igual a la posición del byte siguiente a este byte descartado.
2. Se extrae y descarta la siguiente secuencia de tres bytes (que es **start\_code\_prefix\_one\_3bytes**) del tren de bytes, y la posición actual pasa a ser la posición del siguiente byte después de esta secuencia de tres bytes.
3. Se asigna a **NumBytesInNALunit** el número de bytes comenzando con el byte en la posición actual en el tren de bytes hasta el último byte, inclusive, que precede a alguno de los siguientes elementos:

- a. una secuencia de tres bytes alineada por byte subsiguiente igual a 0x000000, o
  - b. una secuencia de tres bytes alineada por byte subsiguiente igual a 0x000001, o
  - c. el final del tren de bytes, independientemente de cómo se determine éste.
4. Se suprimen los bytes NumBytesInNALunit del tren de bits y se avanza NumBytesInNALunit bytes a la posición actual en el tren. Esta secuencia de bytes es nal\_unit( NumBytesInNALunit ) y se decodifica mediante el proceso de decodificación de unidades NAL.
  5. Si la posición actual en el tren de bytes no se encuentra al final del tren de bytes (no se especifica cómo se determina el final del tren de bytes) y los siguientes bytes del tren de bytes no comienzan con una secuencia de tres bytes igual a 0x000001 y los siguientes bytes del tren de bytes no comienzan con una secuencia de cuatro bytes igual a 0x00000001, el decodificador extrae y descarta todos los elementos sintácticos trailing\_zero\_8bits, haciendo avanzar la posición actual en el tren de bytes, byte por byte, hasta que la posición actual en el tren de bytes sea tal que los siguientes bytes del tren de bytes conformen la secuencia de cuatro bytes 0x00000001 o se llegue al final del tren de bytes (no se especifica cómo se determina que se ha llegado al final del tren de bytes).

### **B.3 Recuperación de la alineación por byte del decodificador (informativo)**

Esta subcláusula no es parte integrante de esta Recomendación | Norma Internacional.

Muchas aplicaciones suministran datos a un decodificador de un modo inherentemente alineado por byte, y por consiguiente no tienen necesidad de utilizar el procedimiento para detectar la alineación de byte orientado a bits que se describe en esta subcláusula.

Se dice que un decodificador está alineado por bytes con un tren de bits cuando el decodificador puede determinar si las posiciones de los datos en el tren de bits están alineadas por bytes. Cuando un decodificador no está alineado por byte con el tren de bytes del codificador, el decodificador puede examinar el tren de bits para buscar el patrón binario '00000000 00000000 00000000 00000001' (31 bits consecutivos iguales a 0 seguidos de un bit igual a 1). El bit justo a continuación de ese patrón es el primer bit de un byte alineado después de un prefijo código de comienzo. Después de detectar este patrón, el decodificador estará alineado por byte con el codificador y posicionado al comienzo de una unidad NAL en el tren de bytes.

Una vez alineado por byte con el codificador, el decodificador podrá examinar el tren de bytes entrante en búsqueda de subsiguientes secuencias de tres bytes 0x000001 y 0x000003.

Cuando se detecta una secuencia de tres bytes 0x000001, significa que se trata de un prefijo código de comienzo.

Cuando se detecta una secuencia de tres bytes 0x000003, el tercer byte (0x03) es un emulation\_prevention\_three\_byte que tiene que descartarse conforme a la subcláusula 7.4.1.

El procedimiento de detección de alineación por byte que se describe en esta subcláusula es funcionalmente equivalente a la búsqueda de una secuencia de tres bytes consecutivos con valor cero (0x000000), sea cual sea la alineación. La detección de este patrón indica que el próximo byte diferente de cero contiene el final de un prefijo código de comienzo (ya que un tren de bytes conforme no puede contener más de 23 bits consecutivos con valor cero sin contener 31 o más bits consecutivos con valor cero, lo que permite la detección de 0x000000 a partir de cualquier posición de alineación inicial), y el primer bit diferente de cero del siguiente byte diferente de cero es el último bit de un byte alineado y es el último bit de un prefijo código de inicio.

## Anexo C

### Decodificador ficticio de referencia

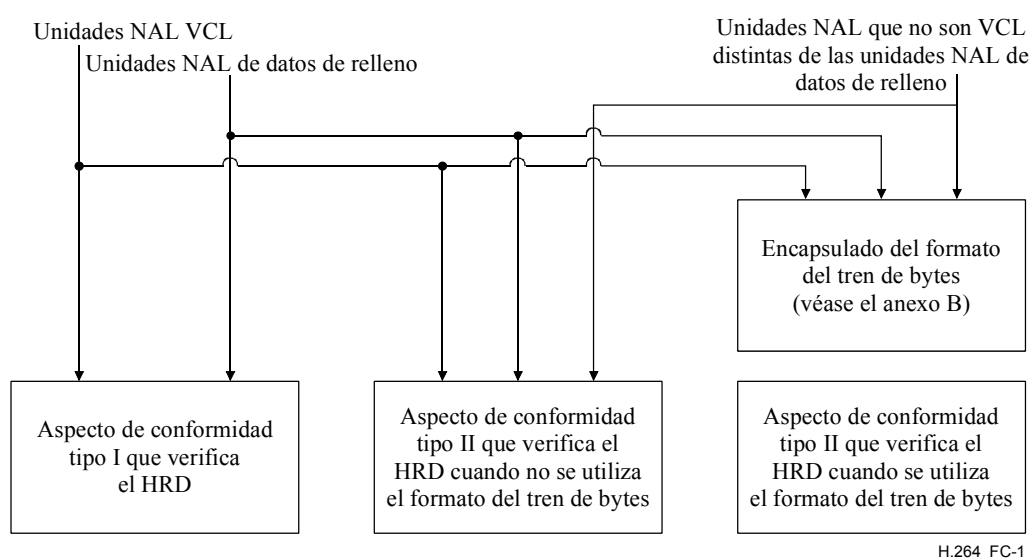
(Este anexo es parte integrante de esta Recomendación | Norma Internacional)

En este anexo se describe el decodificador ficticio de referencia (HRD, *hypothetical reference decoder*) y la manera de utilizarlo para verificar la conformidad del tren de bits y del decodificador.

De acuerdo con esta Recomendación | Norma Internacional hay dos tipos de trenes de bits cuya conformidad verifica el HRD. El primer tipo, denominado tren de bits tipo I, es un tren de unidades NAL que contiene sólo unidades NAL VCL y unidades NAL de datos de relleno de todas las unidades de acceso al tren de bits. El segundo tipo, denominado tren de bits tipo II contiene, además de unidades NAL VCL y unidades NAL de datos de relleno de todas las unidades de acceso en el tren de bits, al menos una de las siguientes:

- unidades NAL adicionales que no son VCL distintas de las unidades NAL de datos de relleno;
- todos los elementos sintácticos `leading_zero_8bits`, `zero_byte`, `start_code_prefix_one_3bytes`, y `trailing_zero_8bits` que forman un tren de bytes a partir del tren de unidades NAL (conforme al anexo B).

En la figura C-1 se indican los tipos de aspectos de conformidad del tren de bits que verifica el HRD.



**Figura C-1 – Estructura de los trenes de bytes y de los trenes de unidades NAL para verificar la conformidad mediante el HRD**

Los elementos sintácticos de las unidades NAL que no son VCL (o los valores por defecto de algunos elementos sintácticos), necesarios para el HRD, se describen en las subcláusulas sobre semántica de la cláusula 7 y de los anexos D y E.

Se utilizan dos tipos de conjuntos de parámetros HRD. Estos conjuntos se indican en la información sobre los posibles usos de vídeo conforme a las subcláusulas E.1 y E.2, que es parte de la estructura sintáctica conjunto de parámetros secuencia.

Para verificar la conformidad de un tren de bits mediante el HRD, se enviarán al HRD todos los conjuntos de parámetros secuencia y de parámetros imagen contenidos en las unidades NAL VCL, y los correspondientes mensajes SEI de periodo de almacenamiento intermedio y de temporización

de imagen correspondientes, con la temporización adecuada, bien sea en el tren de bits (mediante unidades NAL que no son VCL), o por otros medios que no se especifican en esta Recomendación | Norma Internacional.

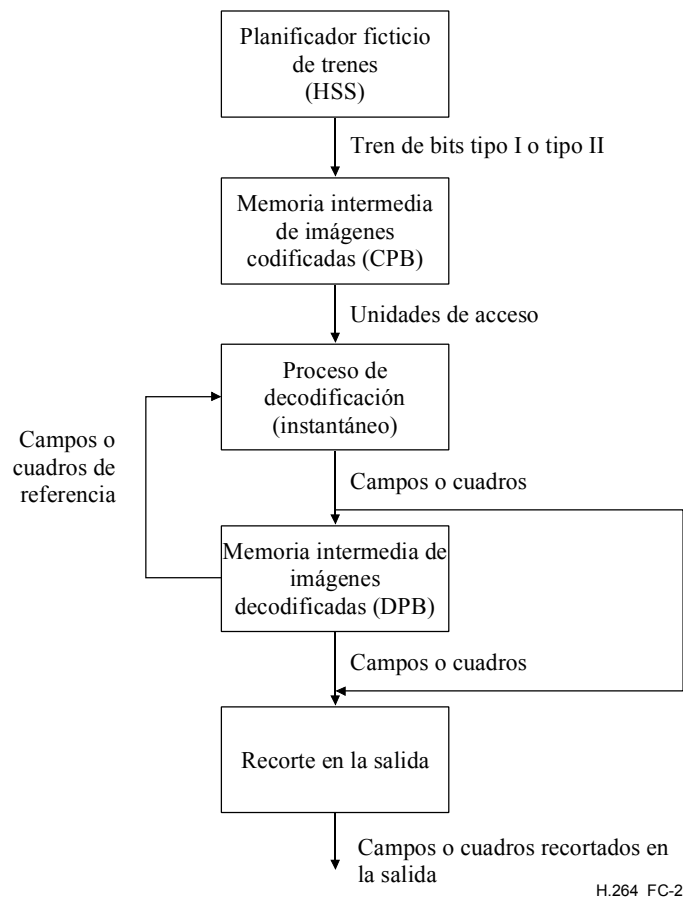
En los anexos C, D y E, la definición de "presencia" de unidades NAL que no son VCL también se cumple cuando esas unidades NAL (o sólo algunas de ellas) se envían a los decodificadores (o al HRD) por otros medios que no se especifican en esta Recomendación | Norma Internacional. Para fines del recuento de bits, sólo se tendrán en consideración los bits apropiados que estén presentes en el tren de bits.

NOTA – Por ejemplo, para lograr la sincronización de una unidad NAL que no es VCL, y que no se envía en el tren de bits con las unidades NAL presentes en el mismo, se podrían indicar dos puntos en el tren de bits, entre los cuales debería estar incluida la unidad NAL que no es VCL, si el codificador hubiese decidido transportarla en el tren de bits.

Cuando el contenido de una unidad NAL que no es VCL se envía a la aplicación por algún medio distinto al tren de bits, no es obligatorio que la representación del contenido de esa unidad utilice la sintaxis especificada en este anexo.

NOTA – Si en el tren de bits se incluye información del HRD, es posible verificar la conformidad de un tren de bits con los requisitos de esta subcláusula basándose únicamente en la información contenida en el tren de bits. Si no se incluye dicha información, que es el caso de todos los trenes de bits tipo I "autónomos", sólo se podrá verificar la conformidad si los datos HRD se suministran por otros medios que no se especifican en esta Recomendación | Norma Internacional.

El HRD contiene una memoria intermedia de imágenes codificadas (CPB), un proceso de decodificación instantánea, una memoria intermedia de imágenes decodificadas (DPB), y un recorte en la salida como se ilustra en la figura C-2.



**Figura C-2 – Modelo de memoria intermedia HRD**

El tamaño de la CPB (número de bits) se especifica en el anexo E mediante  $CpbSize[ SchedSelIdx ]$ . El tamaño de la DPB (número de memorias intermedias de cuadros) se especifica en el anexo E como  $Max( 1, max\_dec\_frame\_buffering)$ .

El HRD funciona de la siguiente manera. El HSS envía los datos correspondientes a las unidades de acceso que fluyen hacia la CPB de acuerdo con un planificador de llegada determinado. Los datos correspondientes a cada unidad de acceso se extraen y decodifican mediante el proceso de decodificación instantáneo en los instantes de extracción de la CPB. Cada imagen decodificada se guarda en la DPB en el momento de extracción de la CPB a menos que se envíe a la salida en ese mismo instante y no sea una imagen de referencia. El instante en que se extrae una imagen de la DPB, después de haberse guardado en ella, es el postrero entre el instante de salida que le corresponde y el instante en que se marca como "imagen no utilizada como referencia".

El funcionamiento de la CPB se describe en la subcláusula C.1. El funcionamiento del decodificador instantáneo se describe en las cláusulas 8 y 9. El funcionamiento de la DPB se describe en la subcláusula C.2. El recorte de salida se describe en la subcláusula C.2.2.

La información de HSS y HRD relativa al número de planificaciones de entrega enumeradas y sus correspondientes velocidades binarias y tamaños de memorias intermedias se describe en las subcláusulas E.1.1, E.1.2, E.2.1 y E.2.2. En las subcláusulas D.1.1 y D.2.1 se describe la inicialización del HRD de acuerdo con las indicaciones contenidas en el mensaje SEI de periodo de almacenamiento en la memoria intermedia. El instante de extracción de las unidades de acceso de la CPB y el momento de salida de la DPB se indican en el mensaje SEI de temporización de imágenes descrito en las subcláusulas D.1.2 y D.2.2. Toda la información de temporización relativa a una determinada unidad de acceso debe llegar antes del instante de extracción de la unidad de acceso de la CPB.

El HRD se utiliza para verificar la conformidad de los trenes de bits y de los decodificadores conforme a las subcláusulas C.3 y C.4, respectivamente.

NOTA – Aunque se garantice la conformidad suponiendo que todas las velocidades de cuadro y los relojes utilizados para generar el tren de bits concuerdan exactamente con los valores indicados en el tren de bits, en un sistema real éstos pueden variar con respecto al valor indicado o especificado.

En este anexo, todos los cálculos aritméticos se efectúan con valores reales, para que no haya propagación de errores de redondeo. Por ejemplo, el número de bits en una CPB justo antes o después de la extracción de una unidad de acceso no es necesariamente un entero.

La variable  $t_c$  se calcula como se indica a continuación y se denomina tic de reloj.

$$t_c = \text{num\_units\_in\_tick} \div \text{time\_scale} \quad (\text{C-1})$$

Para expresar las restricciones en este anexo se define lo siguiente.

- Sea la unidad de acceso  $n$  la  $n$ -ésima unidad de acceso en orden de decodificación, siendo la primera unidad de acceso la unidad 0.
- Sea la imagen  $n$  la imagen codificada primaria o la imagen decodificada primaria de la unidad de acceso  $n$ .

## **C.1 Funcionamiento de la memoria intermedia de imágenes codificadas (CPB)**

Las especificaciones de esta subcláusula se aplican de modo independiente a cada conjunto de parámetros de la CPB que esté incluido y a los puntos de conformidad tipo I y tipo II que se muestran en la figura C-1.

### **C.1.1 Temporización de la llegada del tren de bits**

El HRD puede inicializarse con cualquiera de los mensajes SEI de periodo de almacenamiento intermedio. Antes de la inicialización, la CPB está vacía.



NOTA – Una vez inicializado el HRD no se puede inicializar nuevamente con mensajes SEI de periodo de almacenamiento intermedio subsiguientes.

Se hace referencia a cada unidad de acceso como unidad de acceso  $n$ , donde el número  $n$  identifica la unidad de acceso particular. La unidad de acceso correspondiente al mensaje SEI de periodo de almacenamiento intermedio que inicializa la CPB se denomina unidad de acceso 0. El valor de  $n$  aumenta en una unidad para cada unidad de acceso subsiguiente en orden de decodificación.

El momento en el que llega el primer bit de la unidad de acceso  $n$  a la CPB se conoce como el instante de llegada inicial  $t_{ai}(n)$ .

El instante de llegada inicial de las unidades de acceso se obtiene del modo siguiente.

- Si la unidad de acceso es la unidad 0,  $t_{ai}(0) = 0$ ,
- De lo contrario, (la unidad de acceso es la unidad  $n$ , con  $n > 0$ ), se aplica lo siguiente.
  - Si  $cbr\_flag[ SchedSelIdx ]$  es igual a 1, el instante de llegada inicial de la unidad de acceso  $n$ , es igual al instante de llegada final (que se calcula más adelante) de la unidad de acceso  $n - 1$ , es decir.

$$t_{ai}(n) = t_{af}(n - 1) \quad (C-2)$$

- De lo contrario, ( $cbr\_flag[ SchedSelIdx ]$  es igual a 0), el instante de llegada inicial de la unidad de acceso  $n$  se calcula mediante

$$t_{ai}(n) = \text{Max}( t_{af}(n - 1), t_{ai,earliest}(n) ) \quad (C-3)$$

donde  $t_{ai,earliest}(n)$  se calcula mediante

- Si la unidad de acceso  $n$  no es la primera unidad de acceso de un periodo de almacenamiento subsiguiente,  $t_{ai,earliest}(n)$  se calcula como

$$t_{ai,earliest}(n) = t_{r,n}(n) - (\text{initial\_cpb\_removal\_delay}[ SchedSelIdx ] + \text{initial\_cpb\_removal\_delay\_offset}[ SchedSelIdx ]) \div 90000 \quad (C-4)$$

siendo  $t_{r,n}(n)$  el instante de extracción nominal de la unidad de acceso  $n$  de la CPB conforme a la subcláusula C.1.2 e  $\text{initial\_cpb\_removal\_delay}[ SchedSelIdx ]$  e  $\text{initial\_cpb\_removal\_delay\_offset}[ SchedSelIdx ]$  se indican en el mensaje SEI de periodo de almacenamiento intermedio previo.

- De lo contrario (la unidad de acceso  $n$  es la primera unidad de acceso de un periodo de almacenamiento intermedio subsiguiente),  $t_{ai,earliest}(n)$  se calcula mediante la expresión

$$t_{ai,earliest}(n) = t_{r,n}(n) - (\text{initial\_cpb\_removal\_delay}[ SchedSelIdx ] \div 90000) \quad (C-5)$$

donde  $\text{initial\_cpb\_removal\_delay}[ SchedSelIdx ]$  se especifica en el mensaje SEI de periodo de almacenamiento intermedio de la unidad de acceso  $n$ .

El instante de llegada final de la unidad de acceso  $n$  viene dado por

$$t_{af}(n) = t_{ai}(n) + b(n) \div \text{BitRate}[ SchedSelIdx ] \quad (C-6)$$

siendo  $b(n)$  el tamaño en bits de la unidad de acceso  $n$ , contando los bits de las unidades NAL VCL y las unidades NAL de relleno de datos para el punto de conformidad del tipo I o todos los bits del tren de bits tipo II para el punto de conformidad del tipo II, donde los puntos de conformidad tipo I y tipo II se muestran en la figura C-1.

Para los valores de  $SchedSelIdx$ ,  $\text{BitRate}[ SchedSelIdx ]$ , y de  $\text{CpbSize}[ SchedSelIdx ]$  se establecen las siguientes restricciones.

- Si las unidades de acceso  $n$  y  $n - 1$  son parte de distintas secuencias de vídeo codificado y el contenido de los conjuntos de parámetros secuencia activos de las dos secuencias de vídeo codificado es diferente, el HSS puede elegir un valor  $SchedSelIdx1$  entre los valores de  $SchedSelIdx$  proporcionados por la secuencia de vídeo codificado que contiene la unidad de acceso  $n$  de manera que  $\text{BitRate}[ SchedSelIdx1 ]$  o  $\text{CpbSize}[ SchedSelIdx1 ]$  de la

segunda de las dos secuencias de vídeo codificado (que contiene la unidad de acceso n). El valor de BitRate[ SchedSelIdx1 ] o de CpbSize[ SchedSelIdx1 ] puede ser distinto del valor de BitRate[ SchedSelIdx0 ] o de CpbSize[ SchedSelIdx0 ], siendo SchedSelIdx0 de SchedSelIdx el valor que estaba utilizando la secuencia de vídeo codificado que contiene la unidad de acceso n – 1.

- De lo contrario, el HSS continúa funcionando con los valores anteriores de SchedSelIdx, BitRate[ SchedSelIdx ] y CpbSize[ SchedSelIdx ].

Cuando el HSS selecciona valores de BitRate[ SchedSelIdx ] o CpbSize[ SchedSelIdx ] que difieren de los de la unidad de acceso anterior, se aplica lo siguiente.

- se activa la variable BitRate[ SchedSelIdx ] en el instante  $t_{af}(n)$
- se activa la variable CpbSize[ SchedSelIdx ] de la siguiente manera.
  - si el nuevo valor de CpbSize[ SchedSelIdx ] rebasa el tamaño de la CPB anterior, se activa en el instante  $t_{af}(n)$ ,
  - de lo contrario, se activa el nuevo valor de CpbSize[ SchedSelIdx ] en el instante  $t_r(n)$ .

### C.1.2 Temporización de la extracción de imágenes codificadas

Para la unidad de acceso 0, el instante de extracción nominal de la unidad de acceso de la CPB se calcula mediante:

$$t_{r,n}(0) = \text{initial\_cpb\_removal\_delay}[ \text{SchedSelIdx} ] \div 90000 \quad (\text{C-7})$$

Para la primera unidad de acceso de un periodo de almacenamiento intermedio que no inicializa el HRD, el instante de extracción nominal de la unidad de acceso de la CPB se calcula mediante:

$$t_{r,n}(n) = t_{r,n}(n_b) + t_c * \text{cpb\_removal\_delay}(n) \quad (\text{C-8})$$

donde  $t_{r,n}(n_b)$  es el instante de extracción nominal de la primera unidad de acceso del periodo de almacenamiento intermedio anterior y  $\text{cpb\_removal\_delay}(n)$  es el valor de  $\text{cpb\_removal\_delay}$  que se indica en el mensaje SEI de temporización de imágenes correspondiente a la unidad de acceso n.

Cuando una unidad de acceso n es la primera unidad de un periodo de almacenamiento intermedio,  $n_b$  se pone igual a n en el instante de extracción de la unidad de acceso n.

El instante de extracción nominal  $t_{r,n}(n)$  de una unidad de acceso n que no es la primera unidad de un periodo de almacenamiento intermedio viene dado por:

$$t_{r,n}(n) = t_{r,n}(n_b) + t_c * \text{cpb\_removal\_delay}(n) \quad (\text{C-9})$$

donde  $t_{r,n}(n_b)$  es el instante de extracción de la primera unidad de acceso del periodo de almacenamiento actual y  $\text{cpb\_removal\_delay}(n)$  es el valor de  $\text{cpb\_removal\_delay}$  especificado en el mensaje SEI de temporización asociado a la unidad de acceso n.

El instante de extracción de la unidad de acceso n se define del modo siguiente.

- Si  $\text{low\_delay\_hrd\_flag}$  es igual a 0 o  $t_{r,n}(n) \geq t_{af}(n)$ , el instante de extracción de la unidad de acceso n es:

$$t_r(n) = t_{r,n}(n) \quad (\text{C-10})$$

- De lo contrario ( $\text{low\_delay\_hrd\_flag}$  es igual a 1 y  $t_{r,n}(n) < t_{af}(n)$ ), el instante de extracción de la unidad de acceso n es:

$$t_r(n) = t_{r,n}(n) + t_c * \text{Ceil}((t_{af}(n) - t_{r,n}(n)) \div t_c) \quad (\text{C-11})$$

NOTA – El último caso indica que el tamaño de la unidad de acceso n,  $b(n)$ , es tan grande que resulta imposible la extracción en el instante de extracción nominal.

## C.2 Funcionamiento de la memoria intermedia de imágenes decodificadas (DPB)

La memoria intermedia de imágenes decodificadas contiene memorias intermedias de cuadros. Cada una de estas memorias puede contener un cuadro decodificado, un par de campos complementarios decodificados o un solo campo decodificado (desapareado) marcado como "utilizado como referencia" (imágenes de referencia) o almacenado para salida posterior (imágenes reordenadas o retardadas). Antes de la inicialización, la DPB está vacía (el nivel de ocupación de la DPB se pone a cero). Todos los pasos que se describen en las siguientes subcláusulas de esta cláusula suceden instantáneamente en el momento  $t_r(n)$  y en el orden indicado.

### C.2.1 Decodificación de vacíos en frame\_num y almacenamiento de cuadros "inexistentes"

El proceso de decodificación detecta vacíos en frame\_num, si los hubiese, y los cuadros generados se marcan e insertan en la DPB como se especifica más adelante.

El proceso de decodificación detecta los vacíos en frame\_num y los cuadros generados se marcan conforme a la subcláusula 8.2.5.2.

Después de marcar cada uno de los cuadros generados, cada imagen  $m$  marcada como "no utilizada como referencia" por el proceso de "ventana deslizante" se extrae de la DPB si está marcada también como "inexistente" o si su instante de salida de la DPB es anterior o igual al instante de extracción de la imagen  $n$  actual de la CPB; es decir,  $t_{o,dpb}(m) \leq t_r(n)$ . Cuando se extrae de la DPB un cuadro o el último campo en una memoria intermedia de cuadros, el nivel de ocupación de la DPB disminuye en una unidad. El cuadro generado como "inexistente" se coloca en la DPB y su nivel de ocupación aumenta en una unidad.

### C.2.2 Decodificación e instante de salida de imágenes

La imagen  $n$  se decodifica y su instante de salida de la DPB  $t_{o,dpb}(n)$  se calcula mediante:

$$t_{o,dpb}(n) = t_r(n) + t_c * dpb\_output\_delay(n) \quad (C-12)$$

El instante de salida de la imagen actual se determina de la siguiente manera:

- Si  $t_{o,dpb}(n) = t_r(n)$ , se saca la imagen actual.  
NOTA – Cuando la imagen actual es una imagen de referencia, ésta se almacenará en la DPB.
- De lo contrario ( $t_{o,dpb}(n) > t_r(n)$ ), la imagen actual se almacena en la DPB (conforme a la subcláusula C.2.4) y se saca posteriormente en el instante  $t_{o,dpb}(n)$  a menos que el decodificador o la detección de no\_output\_of\_prior\_pics\_flag igual a 1 en un momento anterior a  $t_{o,dpb}(n)$  indiquen que no debe salir.

La imagen de salida se recortará, utilizando el rectángulo de recorte indicado en el conjunto de parámetros secuencia de la secuencia.

Cuando la imagen  $n$  que ha salido no es la última imagen del tren de bits, el valor de  $\Delta t_{o,dpb}(n)$  se define como:

$$\Delta t_{o,dpb}(n) = t_{o,dpb}(n_n) - t_{o,dpb}(n) \quad (C-13)$$

donde  $n_n$  indica la imagen a continuación de la imagen  $n$  en el orden de salida.

La imagen decodificada se almacena temporalmente (pero no en la DPB).

### C.2.3 Extracción de imágenes de la DPB antes de la posible inserción de la imagen actual

La extracción de imágenes de la DPB antes de la posible inserción de la imagen actual se lleva a cabo de la siguiente manera:

- Si la imagen decodificada es una imagen IDR se aplica lo siguiente.
  - Todas las imágenes de referencia en la DPB se marcan como "no utilizada como referencia" conforme a las subcláusulas 8.2.5.3 y 8.2.5.4.

- Cuando la imagen IDR no es la primera imagen IDR decodificada y el valor de PicWidthInMbs o de FrameHeightInMbs o de max\_dec\_frame\_buffering deducido a partir del conjunto de parámetros secuencia activo difiere, respectivamente, del valor de PicWidthInMbs o de FrameHeightInMbs o de max\_dec\_frame\_buffering deducido a partir del conjunto de parámetros secuencia que estaba activo en la secuencia anterior, el HRD infiere que no\_output\_of\_prior\_pics\_flag es igual a 1, sin tener en cuenta el valor real de no\_output\_of\_prior\_pics\_flag.
  - NOTA – Las implementaciones del decodificador deberían manejar con mayor normalidad que el HRD los cambios de tamaño de los cuadros o de la DPB con relación a los cambios en PicWidthInMbs o FrameHeightInMbs.
- Cuando no\_output\_of\_prior\_pics\_flag es igual a 1 o se infiere que es igual a 1, se vacían todas las memorias intermedias de cuadros en la DPB sin sacar las imágenes que contienen, y el nivel de ocupación de la DPB se pone a 0.
- De lo contrario (la imagen decodificada no es una imagen IDR), se aplica lo siguiente:
  - Si la cabecera de sector de la imagen considerada incluye memory\_management\_control\_operation igual a 5, todas las imágenes de referencia en la DPB se marcan como "no utilizada como referencia".
  - De lo contrario (la cabecera de sector de la imagen considerada no incluye memory\_management\_control\_operation igual a 5), se llama al proceso de marcado de imágenes de referencia decodificadas, especificado en la subcláusula 8.2.5.

Se extraen de la DPB todas las imágenes  $m$ , para las que se cumplen las siguientes condiciones.

- La imagen  $m$  está marcada como "no utilizada como referencia" o la imagen  $m$  no es una imagen de referencia. Cuando una imagen es un cuadro de referencia, se considera que está marcada como "no utilizada como referencia" sólo cuando sus dos campos se han marcado de la misma forma.
- La imagen  $m$  está marcada como "inexistente" o su instante de salida de la DPB es anterior o igual al instante de extracción de la CPB de la imagen  $n$  considerada; es decir,  $t_{o,dpb}(m) \leq t_r(n)$ .

Cuando se extrae de la DPB un cuadro o el último campo de una memoria intermedia de cuadros, el nivel de ocupación de la DPB disminuye en una unidad.

## **C.2.4 Marcado y almacenamiento de la imagen decodificada considerada**

### **C.2.4.1 Marcado y almacenamiento en la DPB de una imagen de referencia decodificada**

Cuando la imagen considerada es una imagen de referencia se almacena en la DPB como se indica a continuación:

- Si la imagen decodificada considerada es el segundo campo (en orden de decodificación) de un par de campos de referencia complementarios, y el primer campo del par aún está en la DPB, la imagen decodificada considerada se almacena en la misma memoria intermedia de cuadros que el primer campo del par.
- De lo contrario, la imagen decodificada considerada se almacena en una memoria intermedia de cuadros vacía, y el nivel de ocupación de la DPB se aumenta en una unidad.

### **C.2.4.2 Almacenamiento en la DPB de una imagen que no es una referencia**

Cuando la imagen considerada no es una imagen de referencia y la imagen  $n$  actual tiene  $t_{o,dpb}(n) > t_r(n)$ , se almacena en la DPB como se indica a continuación:

- Si la imagen decodificada considerada es el segundo campo (en orden de decodificación) de un par de campos que no son referencia complementarios, y el primer campo del par aún

está en la DPB, la imagen decodificada considerada se almacena en la misma memoria intermedia de cuadros que el primer campo del par.

- De lo contrario, la imagen decodificada considerada se almacena en una memoria intermedia de cuadros vacía, y el nivel de ocupación de la DPB aumenta en una unidad.

### C.3 Conformidad del tren de bits

Un tren de bits de datos codificados conforme con esta Recomendación | Norma Internacional cumple con los siguientes requisitos.

El tren de bits está formado de acuerdo con la sintaxis, la semántica y las restricciones especificadas en esta Recomendación | Norma Internacional, aparte de este anexo.

El tren de bits se verifica mediante el HRD como se describe a continuación:

Para los trenes de bits tipo I, el número de pruebas realizadas es igual a  $cpb\_cnt\_minus1 + 1$ , donde  $cpb\_cnt\_minus1$  es el elemento sintáctico `hrd_parameters()` que viene después de `vcl_hrd_parameters_present_flag`, o viene determinado por la aplicación de manera no especificada en esta Recomendación | Norma Internacional. Se realiza una prueba por cada combinación de velocidad binaria y tamaño de CPB especificada por `hrd_parameters()` que viene después de `vcl_hrd_parameters_present_flag`. Cada una de estas pruebas es llevada a cabo en el punto de conformidad tipo I que se muestra en la figura C-1.

Para los trenes de bits tipo II hay dos conjuntos de pruebas. El número de pruebas del primer conjunto es igual a  $cpb\_cnt\_minus1 + 1$ , donde  $cpb\_cnt\_minus1$  es el elemento sintáctico `hrd_parameters()` que viene después de `vcl_hrd_parameters_present_flag`, o viene determinado por la aplicación de manera no especificada en esta Recomendación | Norma Internacional. Se realiza una prueba por cada combinación de velocidad binaria y tamaño de CPB. Cada una de estas pruebas es llevada a cabo en el punto de conformidad tipo I que se muestra en la figura C-1. En estas pruebas, sólo se tendrán en cuenta las unidades VCL y NAL de datos de relleno para la velocidad binaria de entrada y el almacenamiento en la CPB.

El número de pruebas del segundo conjunto, para los trenes de bits del tipo II, es igual a  $cpb\_cnt\_minus1 + 1$ , donde  $cpb\_cnt\_minus1$  es el elemento sintáctico de `hrd_parameters()` a continuación de `nal_hrd_parameters_present_flag`, o se determina mediante la aplicación de otros medios que no se especifican en esta Recomendación | Norma Internacional. Se realiza una prueba por cada combinación de velocidad binaria y tamaño de CPB especificada por `hrd_parameters()` a continuación de `nal_hrd_parameters_present_flag`. Cada una de estas pruebas se lleva a cabo en el punto de conformidad tipo II que se muestra en la figura C-1. En estas pruebas, se tendrán en cuenta todas las unidades NAL (de un tren de unidades NAL tipo II) o todos los bytes (de un tren de bytes) para la velocidad binaria de entrada y el almacenamiento en la CPB.

NOTA – Los parámetros HRD NAL que se fijan mediante un valor de `SchedSelIdx` para el punto de conformidad tipo II que se muestra en la figura C-1 son también suficientes para establecer conformidad HRD VCL en el punto de conformidad tipo I que se muestra en la figura C-1, para valores idénticos de `initial_cpb_removal_delay[ SchedSelIdx ]`, `BitRate[ SchedSelIdx ]` y `CpbSize[ SchedSelIdx ]` en el caso de tasa de bits variable (VBR) (`cbr_flag[ SchedSelIdx ]` igual a 0). Esto se debe a que el flujo de datos que ingresa al punto de conformidad tipo I es un subconjunto del flujo de datos que ingresa al punto de conformidad tipo II y porque, en el caso de VBR, se permite que el CPB se vacíe y permanezca vacío hasta el instante en que está previsto que empiece a llegar la siguiente imagen. Por ejemplo cuando se asignan parámetros HRD NAL para el punto de conformidad tipo II que no sólo caen dentro de los límites establecidos para los parámetros HRD NAL para la conformidad de perfil del ítem j de la subcláusula A.3.1, sino que caen dentro de los límites establecidos para los parámetros HRD VCL para la conformidad de perfil del ítem i de la subcláusula A.3.1, se garantiza también que la conformidad del HRD VCL para el punto de conformidad tipo I cae dentro de los límites del ítem i de la subcláusula A.3.1.

En cada una de las pruebas se deben cumplir todas las condiciones que se indican a continuación para los trenes de bits conformantes.

- Para cada unidad de acceso  $n$ , con  $n > 0$ , correspondiente a un mensaje SEI de periodo de almacenamiento intermedio,  $\Delta t_{g,90}(n)$  viene dado por:

$$\Delta t_{g,90}(n) = 90000 * (t_{r,n}(n) - t_{af}(n-1)) \quad (C-14)$$

El valor de `initial_cpb_removal_delay[ SchedSelIdx ]` se limitará como sigue.

- Si `cbr_flag[ SchedSelIdx ]` es igual a 0,

$$\text{initial\_cpb\_removal\_delay[ SchedSelIdx ]} \leq \text{Ceil}(\Delta t_{g,90}(n)) \quad (C-15)$$

- De lo contrario, (`cbr_flag[ SchedSelIdx ]` es igual a 1),

$$\text{Floor}(\Delta t_{g,90}(n)) \leq \text{initial\_cpb\_removal\_delay[ SchedSelIdx ]} \leq \text{Ceil}(\Delta t_{g,90}(n)) \quad (C-16)$$

NOTA – El número exacto de bits en el CPB en el instante de extracción de cada imagen puede depender del mensaje SEI del periodo de almacenamiento que se seleccione para inicializar el HRD. Los codificadores deben tener en cuenta esto para garantizar que se cumplan todas las restricciones especificadas sin importar el mensaje SEI del periodo de almacenamiento que se seleccione para inicializar el HRD, ya que cualquiera de los mensajes SEI de periodo de almacenamiento intermedio puede inicializar el HRD.

- Un desbordamiento de la CPB se define como una condición en la que el número total de bits en la CPB es mayor que el tamaño de la CPB. La CPB nunca ha de desbordarse.
- Una condición de subutilización de la CPB se define como aquella en la que  $t_{r,n}(n)$  es menor que  $t_{af}(n)$ . Si `low_delay_hrd_flag` es igual a 0, la CPB nunca estará subutilizada.
- Los instantes de extracción nominal de imágenes de la CPB (comenzando a partir de la segunda imagen en el orden de decodificación) deben cumplir las restricciones de  $t_{r,n}(n)$  y  $t_r(n)$  indicadas en las subcláusulas A.3.1 y A.3.2 para el perfil y nivel especificados en el tren de bits.
- Inmediatamente después de añadir una imagen decodificada en la DPB, el nivel de ocupación de la DPB debe ser menor que o igual al tamaño de la DPB de acuerdo con las restricciones indicadas en los anexos A, D y E para el perfil y nivel especificados en el tren de bits.
- Todas las imágenes de referencia deben estar en la DPB cuando se necesiten para la predicción. Cada imagen estará en la DPB en su instante de salida de la DPB a menos que no esté almacenada en ella, o que haya sido extraída de la DPB antes de su instante de salida mediante uno de los procesos especificados en la subcláusula C.2.
- El valor de  $\Delta t_{o,dpb}(n)$  dado por la ecuación C-13, que representa la diferencia entre el instante de salida de una imagen y el de la imagen inmediatamente siguiente en el orden de salida, debe satisfacer la restricción indicada en la subcláusula A.3.1 para el perfil y el nivel especificados en el tren de bits.

#### C.4 Conformidad del decodificador

Un decodificador conforme con esta Recomendación | Norma Internacional cumple con los siguientes requisitos.

Un decodificador que sea conforme con un determinado perfil y nivel decodificará satisfactoriamente todos los trenes de bits que sean conformes según lo especificado en la subcláusula C.3, para la conformidad del decodificador, siempre que el decodificador reciba oportunamente todos los conjuntos de parámetros secuencia y los conjuntos de parámetros imagen contenidos en las unidades NAL VCL, y los mensajes SEI de periodo de almacenamiento intermedio y de temporización de imágenes apropiados, ya sea en el tren de bits (mediante unidades

NAL que no son VCL), o por medios externos que no se especifican en esta Recomendación | Norma Internacional.

Hay dos tipos de conformidad que puede tener un decodificador: conformidad de temporización de salida y conformidad de orden de salida.

Para verificar la conformidad de un decodificador, los trenes de bits de prueba conformes al perfil y nivel indicados, que se describen en la subcláusula C.3, se hacen pasar por el HRD y por el decodificador en prueba (DUT) mediante un planificador ficticio de trenes (HSS). Todas las imágenes resultantes del HRD serán también imágenes resultantes del DUT y, para cada imagen resultante del HRD, los valores de todas las muestras resultantes del DUT para la imagen correspondiente serán iguales a los valores de las muestras resultantes del HRD.

Para comprobar la conformidad de la temporización de salida del decodificador, el HSS funciona como se describió anteriormente, y de modo que las planificaciones de entrega seleccionadas sean del subconjunto de valores de SchedSelIdx para el que la velocidad binaria y el tamaño de la CPB se restringen conforme al anexo A, para el perfil y nivel especificados, o con planificaciones de entrega "interpoladas", que se describen más adelante, para las cuales la velocidad binaria y el tamaño de la CPB se restringen conforme al anexo A. Se utiliza la misma planificación de entrega para HRD y DUT.

Cuando se incluyen los parámetros HRD y los mensajes SEI de periodo de almacenamiento intermedio con  $cpb\_cnt\_minus1$  mayor que 0, el decodificador podrá decodificar el tren de bits tal y como los recibe del HSS que funciona utilizando un planificador de entrega "interpolado" especificado con una velocidad binaria de pico  $r$ , tamaño de CPB  $c(r)$ , y un retardo inicial de extracción de la CPB ( $f(r) \div r$ ) del modo siguiente:

$$\alpha = (r - \text{BitRate}[\text{SchedSelIdx} - 1]) \div (\text{BitRate}[\text{SchedSelIdx}] - \text{BitRate}[\text{SchedSelIdx} - 1]), \quad (\text{C-17})$$

$$c(r) = \alpha * \text{CpbSize}[\text{SchedSelIdx}] + (1 - \alpha) * \text{CpbSize}[\text{SchedSelIdx} - 1], \quad (\text{C-18})$$

$$f(r) = \alpha * \text{initial\_cpb\_removal\_delay}[\text{SchedSelIdx}] * \text{BitRate}[\text{SchedSelIdx}] + (1 - \alpha) * \text{initial\_cpb\_removal\_delay}[\text{SchedSelIdx} - 1] * \text{BitRate}[\text{SchedSelIdx} - 1] \quad (\text{C-19})$$

para todo  $\text{SchedSelIdx} > 0$  y  $r$  que cumpla  $\text{BitRate}[\text{SchedSelIdx} - 1] \leq r \leq \text{BitRate}[\text{SchedSelIdx}]$  y de manera que  $r$  y  $c(r)$  estén dentro de los límites especificados en el anexo A para la velocidad binaria y el tamaño de memoria intermedia máximos para el perfil y nivel especificados.

NOTA –  $\text{initial\_cpb\_removal\_delay}[\text{SchedSelIdx}]$  puede ser diferente de un periodo de almacenamiento intermedio a otro y tendrá que recalcularse.

Para comprobar la conformidad de la temporización de salida del decodificador, se utiliza un HRD como el descrito anteriormente y la temporización (relativa al tiempo de entrega del primer bit) para la salida de la imagen es la misma para HRD y DUT, salvo un retardo constante.

Para comprobar la conformidad del orden de salida del decodificador, el HSS entrega el tren de bits al DUT "por demanda" de éste, lo que significa que el HSS entrega bits (en el orden de decodificación) únicamente cuando el DUT necesita más bits para continuar con su procesamiento. Se utiliza un HRD como se describe más adelante, y el HSS entrega el tren de bits al HRD mediante una de las planificaciones especificadas en el tren de bits o a través de una planificación "interpolada" de manera que la velocidad binaria y el tamaño de la CPB se restrinjan como se especifica en el anexo A. El orden de la salida de las imágenes será el mismo para HRD y DUT.

NOTA – Esto significa que en esta prueba, la memoria intermedia de imágenes codificadas del DUT podría ser tan pequeña como el tamaño de la unidad de acceso más grande.

Para la conformidad del orden de salida del decodificador, el tamaño de la CPB del HRD es igual a  $\text{CpbSize}[\text{SchedSelIdx}]$  de la planificación seleccionada y el tamaño de la DPB es igual a  $\text{MaxDpbSize}$ . El instante de extracción de la CPB para el HRD es igual al instante de llegada del

último bit y la decodificación se lleva a cabo inmediatamente. El funcionamiento de la DPB de este HRD se describe más adelante.

#### **C.4.1 Funcionamiento de la DPB de orden de salida**

La memoria intermedia de imágenes decodificadas contiene memorias intermedias de cuadros. Cada una de éstas puede contener un cuadro decodificado, un par de campos complementarios decodificados o un solo campo decodificado (desapareado) que está marcado como "utilizado como referencia" o se almacena para salida posterior (imágenes reordenadas). El nivel de ocupación de la DPB, medida en cuadros, durante la inicialización del HRD, se pone a 0. Todos los pasos que se describen a continuación suceden instantáneamente cuando se extrae una unidad de acceso de la CPB, y en el orden indicado.

#### **C.4.2 Decodificación de vacíos en frame\_num y almacenamiento de imágenes "inexistentes"**

Si ha lugar, el proceso de decodificación detecta los vacíos en frame\_num y se infiere el número necesario de cuadros "inexistentes" en el orden especificado por la generación de valores de UnusedShortTermFrameNum en la ecuación 7-10, como se especifica en la subcláusula 8.2.5.2. Cada cuadro "inexistente" se almacena en el DPB de la manera siguiente.

- Si no hay ninguna memoria intermedia de cuadro vacía (es decir, el nivel de ocupación de la DPB es igual al tamaño de la DPB), se invoca repetidamente el proceso de "vaciado" especificado en la subcláusula C.4.5.3 hasta que haya una memoria intermedia desocupada en la cual se pueda almacenar el cuadro "inexistente".
- El cuadro "inexistente" se almacena en una memoria intermedia vacía y se marca como "no necesario para la salida", y el nivel de ocupación de la DPB se incrementa en uno.

#### **C.4.3 Decodificación de imágenes**

La imagen codificada primaria n se decodifica y se almacena temporalmente (pero no en la DPB).

#### **C.4.4 Extracción de imágenes de la DPB antes de la posible inserción de la imagen actual**

La extracción de imágenes de la DPB antes de la posible inserción de la imagen actual se lleva a cabo como se indica a continuación.

- Si la imagen decodificada es una imagen IDR se aplica lo siguiente.
  - Todas las imágenes de referencia en la DPB se marcan como "no utilizadas como referencia" conforme a la subcláusula 8.2.5.
  - Cuando la imagen IDR no es la primera imagen IDR decodificada y el valor de PicWidthInMbs o de FrameHeightInMbs o de max\_dec\_frame\_buffering deducido a partir del conjunto de parámetros secuencia activos difiere, respectivamente, del valor de PicWidthInMbs o de FrameHeightInMbs o de max\_dec\_frame\_buffering deducido a partir del conjunto de parámetros secuencia que estaba activo en la secuencia anterior el HRD infiere que no\_output\_of\_prior\_pics\_flag es igual a 1, sin tener en cuenta el valor real de no\_output\_of\_prior\_pics\_flag.

NOTA – Las implementaciones del decodificador deben manejar con mayor normalidad que el HRD los cambios de tamaño de los cuadros o de la DPB con relación a los cambios de PicWidthInMbs o de FrameHeightInMbs.

- Cuando no\_output\_of\_prior\_pics\_flag es igual a 1 o se infiere que es igual a 1, se vacían todas las memorias intermedias de cuadros en la DPB sin sacar las imágenes que contienen, y el nivel de ocupación de la DPB se pone a 0.
- De lo contrario (la imagen decodificada no es una imagen IDR) se invoca el proceso de marcado de imágenes de referencia decodificadas conforme a la subcláusula 8.2.5. Se vacían (sin generar una salida) las memorias intermedias de cuadros que contienen un cuadro o un par de campos complementarios o un campo desapareado que están marcados



como "no necesario para la salida" y "no utilizado como referencia" , y el nivel de ocupación de la DPB disminuye en un número igual al de memorias intermedias de cuadros que se desocuparon.

Cuando la imagen considerada es una imagen IDR y `no_output_of_prior_pics_flag` es distinto de 1 y no se infiere que sea igual a 1, o la imagen considerada tiene `memory_management_control_operation` igual a 5, todas las memorias intermedias de cuadros en la DPB que no están vacías se vacían llamando reiteradamente al proceso de "vaciado " especificado en la subcláusula C.4.5.3, y el nivel de ocupación de la DPB se pone a 0.

#### **C.4.5 Marcado y almacenamiento de la imagen decodificada considerada**

##### **C.4.5.1 Marcado y almacenamiento en la DPB de una imagen de referencia decodificada**

Cuando la imagen considerada es una imagen de referencia, se almacena en la DPB como sigue.

- Si la imagen decodificada considerada es el segundo campo (en orden de decodificación) de un par de campos de referencia complementarios, y el primer campo del par está aún en la DPB, la imagen considerada se almacena en la misma memoria intermedia de cuadros que el primer campo del par.
- De lo contrario, se realizan las siguientes operaciones:
  - Si no hay memoria intermedia de cuadros vacía (es decir, el nivel de ocupación de la DPB es igual al tamaño de la DPB), se invoca repetidamente el proceso de "vaciado" descrito en la subcláusula C.4.5.3 hasta que haya una memoria intermedia vacía en la cual almacenar la imagen decodificada actual.
  - La imagen decodificada considerada se almacena en una memoria intermedia de cuadros vacía y se marca como "necesaria para la salida", y el nivel de ocupación de la DPB se aumenta en una unidad.

##### **C.4.5.2 Marcado y almacenamiento en la DPB de una imagen decodificada que no es de referencia**

Cuando la imagen considerada no es una imagen de referencia, se llevan a cabo las siguientes operaciones.

- Si la imagen decodificada considerada es el segundo campo (en el orden de decodificación) de un par de campos complementarios que no son de referencia y el primer campo del par está aún en la DPB, la imagen considerada se almacena en la misma memoria intermedia de cuadros que el primer campo del par.
- De lo contrario, se realizan las siguientes operaciones de manera repetitiva hasta que la imagen decodificada actual se haya recortado y sacado o haya sido almacenada en la DPB:
  - Si no hay memoria intermedia de cuadros vacía (es decir, el nivel de ocupación de la DPB es igual al tamaño de la DPB), se aplica lo siguiente:
    - Si la imagen considerada no tiene un valor de `PicOrderCnt()` menor que todas las imágenes en la DPB que están marcadas como "necesaria para la salida", se ejecuta el proceso de "vaciado" descrito en la subcláusula C.4.5.3.
    - De lo contrario (la imagen actual tiene un valor de `PicOrderCnt()` menor que todas las imágenes de la DPB que están marcadas como "necesaria para la salida"), se recorta la imagen considerada, utilizando el rectángulo de recorte especificado en el conjunto de parámetros secuencia de la secuencia y se saca la imagen recortada.
  - De lo contrario (hay una memoria intermedia vacía, es decir, el nivel de ocupación de la DPB es menor que el tamaño de la DPB) la imagen decodificada considerada se almacena en una memoria intermedia de cuadros vacía y se marca como "necesaria para la salida", y el nivel de ocupación de la DPB se aumenta en una unidad.

### C.4.5.3 Proceso de "vaciado"

El proceso de "vaciado" se invoca en los siguientes casos.

- No hay ninguna memoria intermedia de cuadro vacía (es decir, el nivel de ocupación de la DPB es igual al tamaño de la DPB) y se requiere una memoria intermedia vacía para el almacenamiento de un cuadro "inexistente" inferido, como se describe en la subcláusula C.4.2.
- La imagen actual es una imagen IDR y `no_output_of_prior_pics_flag` no es igual a 1 ni se infiere que sea igual a 1, como se describe en la subcláusula C.4.4.
- La imagen actual tiene `memory_management_control_operation` igual a 5, como se describe en la subcláusula C.4.4.
- No hay ninguna memoria intermedia vacía (es decir, el nivel de ocupación de la DPB es igual al tamaño de la DPB) y se requiere una memoria intermedia vacía para el almacenamiento de una imagen de referencia (no IDR) decodificada, como se describe en la subcláusula C.4.5.1.
- No hay ninguna memoria intermedia vacía (es decir, el nivel de ocupación de DPB es igual al tamaño de DPB) y la imagen considerada es una imagen que no es de referencia y que no es el segundo campo de un par de campos complementarios que no son de referencia y hay imágenes en la DPB que están marcadas como "necesaria para salida" que anteceden a la imagen actual, que no es de referencia, en el orden de salida, como se describe en la subcláusula C.4.5.2, por lo que se requiere una memoria intermedia vacía para almacenar la imagen en consideración.

El proceso de "vaciado" consiste en lo siguiente:

- La imagen o el par de campos de referencia complementarios que deben salir primero se seleccionan de la forma siguiente:
  - Se selecciona la memoria intermedia de cuadro que contiene la imagen cuyo valor de `PicOrderCnt()` es el más pequeño entre todas las imágenes de la DPB que están marcadas como "necesaria para salida".
  - Si esta memoria intermedia de cuadro contiene un par de campos complementarios que no son de referencia con ambos campos marcados como "necesario para salida" y ambos campos tienen el mismo `PicOrderCnt()`, se selecciona para salir primero, el primero de los dos campos, según el orden de decodificación.
  - De lo contrario si esta memoria intermedia de cuadro contiene un par de campos de referencia complementarios, ambos marcados como "necesario para salida" y ambos con el mismo `PicOrderCnt()`, se selecciona para salir primero la totalidad del par de campos complementarios de referencia.
  - De lo contrario, se selecciona para salir primero la imagen de esta memoria intermedia de cuadro cuyo valor de `PicOrderCnt()` es el más pequeño.
- Si una sola imagen se considera como primera para la salida, la imagen se recorta utilizando el rectángulo de recorte especificado en el conjunto de parámetros de secuencia para la secuencia, la imagen recortada es sacada y la imagen se marca como "no necesaria para salida".
- De lo contrario (un par de campos de referencia complementarios son considerados como primeros para la salida), los dos campos del par de campos de referencia complementarios se recortan utilizando el rectángulo de recorte especificado en el conjunto de parámetros de secuencia para la secuencia, los dos campos del par de campos de referencia complementarios se sacan al mismo tiempo, y ambos campos del par de campos de referencia complementarios se marcan como "no necesario para salida".

- Se revisa la memoria intermedia del cuadro que contenía la imagen o el par de campos de referencia complementarios que fueron recortados y sacados, y si se cumple alguna de las condiciones siguientes, se vacía la memoria intermedia del cuadro y el nivel de ocupación de la DPB se disminuye en 1.
  - La memoria intermedia de cuadro contiene un campo desapareado que no es de referencia.
  - La memoria intermedia de cuadro contiene un cuadro que no es de referencia.
  - La memoria intermedia de cuadro contiene un par de campos complementarios que no son de referencia, ambos marcados como "no necesario para salida".
  - La memoria intermedia de cuadro contiene un campo de referencia desapareado marcado como "no utilizado como referencia".
  - La memoria intermedia de cuadro contiene un cuadro de referencia con los dos campos marcados como "no utilizado como referencia".
  - La memoria intermedia de cuadro contiene un par de campos de referencia complementarios, ambos marcados como "utilizado como referencia" y "no necesario para salida".

## **Anexo D**

### **Información de perfeccionamiento complementaria**

(Este anexo es parte integrante de esta Recomendación | Norma Internacional)

En este anexo se especifica la sintaxis y la semántica de las cabidas útiles del mensaje SEI.

Los mensajes SEI son útiles en los procesos relacionados con la decodificación, visualización y para otras finalidades. No obstante, los mensajes SEI no son necesarios para la reconstrucción de las muestras luma o croma mediante el proceso de decodificación. Los decodificadores conformes no necesitan procesar esta información para obtener la conformidad del orden de salida con esta Recomendación | Norma Internacional (para la definición de conformidad véase el anexo C). La información del mensaje SEI es necesaria en parte, para verificar la conformidad del tren de bits y la conformidad de la temporización de salida del decodificador.

La especificación de la presencia de los mensajes SEI del anexo D, se satisface también cuando esos mensajes (o algún subconjunto de ellos) llegan a los decodificadores (o al HRD) por otros medios que no se especifican en esta Recomendación | Norma Internacional. Cuando el tren de bits contenga los mensajes SEI, estos obedecerán a la sintaxis y a la semántica descritas en las subcláusulas 7.3.2.3 y 7.4.2.3 y en este anexo. Cuando el contenido de un mensaje SEI llegue a la aplicación por otros medios distintos del tren de bits, no es obligatorio que la representación del contenido del mensaje SEI utilice la sintaxis descrita en este anexo. Para fines del recuento de bits, solo se han de tener en cuenta los bits apropiados que contiene realmente el tren de bits.

## D.1 Sintaxis de la cabida útil SEI

	C	Descriptor
sei_payload( payloadType, payloadSize ) {		
if( payloadType == 0 )		
buffering_period( payloadSize )	5	
else if( payloadType == 1 )		
pic_timing( payloadSize )	5	
else if( payloadType == 2 )		
pan_scan_rect( payloadSize )	5	
else if( payloadType == 3 )		
filler_payload( payloadSize )	5	
else if( payloadType == 4 )		
user_data_registered_itu_t35( payloadSize )	5	
else if( payloadType == 5 )		
user_data_unregistered( payloadSize )	5	
else if( payloadType == 6 )		
recovery_point( payloadSize )	5	
else if( payloadType == 7 )		
dec_ref_pic_marking_repetition( payloadSize )	5	
else if( payloadType == 8 )		
spare_pic( payloadSize )	5	
else if( payloadType == 9 )		
scene_info( payloadSize )	5	
else if( payloadType == 10 )		
sub_seq_info( payloadSize )	5	
else if( payloadType == 11 )		
sub_seq_layer_characteristics( payloadSize )	5	
else if( payloadType == 12 )		
sub_seq_characteristics( payloadSize )	5	
else if( payloadType == 13 )		
full_frame_freeze( payloadSize )	5	
else if( payloadType == 14 )		
full_frame_freeze_release( payloadSize )	5	
else if( payloadType == 15 )		
full_frame_snapshot( payloadSize )	5	
else if( payloadType == 16 )		
progressive_refinement_segment_start( payloadSize )	5	
else if( payloadType == 17 )		
progressive_refinement_segment_end( payloadSize )	5	
else if( payloadType == 18 )		
motion_constrained_slice_group_set( payloadSize )	5	
else		
reserved_sei_message( payloadSize )	5	
if( !byte_aligned( ) ) {		
<b>bit_equal_to_one</b> /* igual a 1 */	5	f(1)
while( !byte_aligned( ) )		
<b>bit_equal_to_zero</b> /* igual a 0 */	5	f(1)
}		
}		

### D.1.1 Sintaxis del mensaje SEI de periodo de almacenamiento intermedio

	<b>C</b>	<b>Descriptor</b>
buffering_period( payloadSize ) {		
<b>seq_parameter_set_id</b>	5	ue(v)
if( NalHrdBpPresentFlag ) {		
for( SchedSelIdx = 0; SchedSelIdx <= cpb_cnt_minus1; SchedSelIdx++ ) {		
<b>initial_cpb_removal_delay</b> [ SchedSelIdx ]	5	u(v)
<b>initial_cpb_removal_delay_offset</b> [ SchedSelIdx ]	5	u(v)
}		
}		
if( VclHrdBpPresentFlag ) {		
for( SchedSelIdx = 0; SchedSelIdx <= cpb_cnt_minus1; SchedSelIdx++ ) {		
<b>initial_cpb_removal_delay</b> [ SchedSelIdx ]	5	u(v)
<b>initial_cpb_removal_delay_offset</b> [ SchedSelIdx ]	5	u(v)
}		
}		
}		

## D.1.2 Sintaxis del mensaje SEI de temporización de imágenes

	C	Descriptor
pic_timing( payloadSize ) {		
if( CpbDpbDelaysPresentFlag ) {		
<b>cpb_removal_delay</b>	5	u(v)
<b>dpb_output_delay</b>	5	u(v)
}		
if( pic_struct_present_flag ) {		
<b>pic_struct</b>	5	u(4)
for( i = 0; i < NumClockTS ; i++ ) {		
<b>clock_timestamp_flag[ i ]</b>	5	u(1)
if( clock_timestamp_flag[i] ) {		
<b>ct_type</b>	5	u(2)
<b>nuit_field_based_flag</b>	5	u(1)
<b>counting_type</b>	5	u(5)
<b>full_timestamp_flag</b>	5	u(1)
<b>discontinuity_flag</b>	5	u(1)
<b>cnt_droppeded_flag</b>	5	u(1)
<b>n_frames</b>	5	u(8)
if( full_timestamp_flag ) {		
<b>seconds_value</b> /* 0..59 */	5	u(6)
<b>minutes_value</b> /* 0..59 */	5	u(6)
<b>hours_value</b> /* 0..23 */	5	u(5)
} else {		
<b>seconds_flag</b>	5	u(1)
if( seconds_flag ) {		
<b>seconds_value</b> /* range 0..59 */	5	u(6)
<b>minutes_flag</b>	5	u(1)
if( minutes_flag ) {		
<b>minutes_value</b> /* 0..59 */	5	u(6)
<b>hours_flag</b>	5	u(1)
if( hours_flag )		
<b>hours_value</b> /* 0..23 */	5	u(5)
}		
}		
}		
}		
if( time_offset_length > 0 )		
<b>time_offset</b>	5	i(v)
}		
}		
}		

### D.1.3 Sintaxis del mensaje SEI de rectángulo de selección

	<b>C</b>	<b>Descriptor</b>
pan_scan_rect( payloadSize ) {		
pan_scan_rect_id	5	ue(v)
pan_scan_rect_cancel_flag	5	u(1)
if( !pan_scan_rect_cancel_flag ) {		
pan_scan_cnt_minus1	5	ue(v)
for( i = 0; i <= pan_scan_cnt_minus1; i++ ) {		
pan_scan_rect_left_offset[ i ]	5	se(v)
pan_scan_rect_right_offset[ i ]	5	se(v)
pan_scan_rect_top_offset[ i ]	5	se(v)
pan_scan_rect_bottom_offset[ i ]	5	se(v)
}		
pan_scan_rect_repetition_period	5	ue(v)
}		
}		

### D.1.4 Sintaxis del mensaje SEI de cabida útil de relleno

	<b>C</b>	<b>Descriptor</b>
filler_payload( payloadSize ) {		
for( k = 0; k < payloadSize; k++ )		
ff_byte /* igual a 0xFF */	5	f(8)
}		

### D.1.5 Sintaxis del mensaje SEI de datos de usuario registrados conforme a la Rec. UIT-T T.35

	<b>C</b>	<b>Descriptor</b>
user_data_registered_itu_t_t35( payloadSize ) {		
itu_t_t35_country_code	5	b(8)
if( itu_t_t35_country_code != 0xFF )		
i = 1		
else {		
itu_t_t35_country_code_extension_byte	5	b(8)
i = 2		
}		
do {		
itu_t_t35_payload_byte	5	b(8)
i++		
} while( i < payloadSize )		
}		

### D.1.6 Sintaxis del mensaje SEI de datos de usuario no registrados

	<b>C</b>	<b>Descriptor</b>
user_data_unregistered( payloadSize ) {		
<b>uuid_iso_iec_11578</b>	5	u(128)
for( i = 16; i < payloadSize; i++ )		
<b>user_data_payload_byte</b>	5	b(8)
}		

### D.1.7 Sintaxis del mensaje SEI de punto de recuperación

	<b>C</b>	<b>Descriptor</b>
recovery_point( payloadSize ) {		
<b>recovery_frame_cnt</b>	5	ue(v)
<b>exact_match_flag</b>	5	u(1)
<b>broken_link_flag</b>	5	u(1)
<b>changing_slice_group_idc</b>	5	u(2)
}		

### D.1.8 Sintaxis del mensaje SEI de repetición de marcado de imágenes de referencia decodificadas

	<b>C</b>	<b>Descriptor</b>
dec_ref_pic_marking_repetition( payloadSize ) {		
<b>original_idr_flag</b>	5	u(1)
<b>original_frame_num</b>	5	ue(v)
if( !frame_mbs_only_flag ) {		
<b>original_field_pic_flag</b>	5	u(1)
if( original_field_pic_flag )		
<b>original_bottom_field_flag</b>	5	u(1)
}		
dec_ref_pic_marking( )	5	
}		



### D.1.9 Sintaxis del mensaje SEI de imágenes de reserva

	<b>C</b>	<b>Descriptor</b>
spare_pic( payloadSize ) {		
<b>target_frame_num</b>	5	ue(v)
<b>spare_field_flag</b>	5	u(1)
if( spare_field_flag )		
<b>target_bottom_field_flag</b>	5	u(1)
<b>num_spare_pics_minus1</b>	5	ue(v)
for( i = 0; i < num_spare_pics_minus1 + 1; i++ ) {		
<b>delta_spare_frame_num[ i ]</b>	5	ue(v)
if( spare_field_flag )		
<b>spare_bottom_field_flag[ i ]</b>	5	u(1)
<b>spare_area_idc[ i ]</b>	5	ue(v)
if( spare_area_idc[ i ] == 1 )		
for( j = 0; j < PicSizeInMapUnits; j++ )		
<b>spare_unit_flag[ i ][ j ]</b>	5	u(1)
else if( spare_area_idc[ i ] == 2 ) {		
mapUnitCnt = 0		
for( j=0; mapUnitCnt < PicSizeInMapUnits; j++ ) {		
<b>zero_run_length[ i ][ j ]</b>	5	ue(v)
mapUnitCnt += zero_run_length[ i ][ j ] + 1		
}		
}		
}		
}		
}		

### D.1.10 Sintaxis del mensaje SEI de información de escena

	<b>C</b>	<b>Descriptor</b>
scene_info( payloadSize ) {		
<b>scene_info_present_flag</b>	5	u(1)
if( scene_info_present_flag ) {		
<b>scene_id</b>	5	ue(v)
<b>scene_transition_type</b>	5	ue(v)
if( scene_transition_type > 3 )		
<b>second_scene_id</b>	5	ue(v)
}		
}		

### D.1.11 Sintaxis del mensaje SEI de información de subsecuencia

	<b>C</b>	<b>Descriptor</b>
sub_seq_info( payloadSize ) {		
<b>sub_seq_layer_num</b>	5	ue(v)
<b>sub_seq_id</b>	5	ue(v)
<b>first_ref_pic_flag</b>	5	u(1)
<b>leading_non_ref_pic_flag</b>	5	u(1)
<b>last_pic_flag</b>	5	u(1)
<b>sub_seq_frame_num_flag</b>	5	u(1)
if( sub_seq_frame_num_flag )		
<b>sub_seq_frame_num</b>	5	ue(v)
}		

### D.1.12 Sintaxis del mensaje SEI de características de la capa subsecuencia

	<b>C</b>	<b>Descriptor</b>
sub_seq_layer_characteristics( payloadSize ) {		
<b>num_sub_seq_layers_minus1</b>	5	ue(v)
for( layer = 0; layer <= num_sub_seq_layers_minus1; layer++ ) {		
<b>accurate_statistics_flag</b>	5	u(1)
<b>average_bit_rate</b>	5	u(16)
<b>average_frame_rate</b>	5	u(16)
}		
}		

### D.1.13 Sintaxis del mensaje SEI de características de subsecuencia

	<b>C</b>	<b>Descriptor</b>
sub_seq_characteristics( payloadSize ) {		
<b>sub_seq_layer_num</b>	5	ue(v)
<b>sub_seq_id</b>	5	ue(v)
<b>duration_flag</b>	5	u(1)
if( duration_flag )		
<b>sub_seq_duration</b>	5	u(32)
<b>average_rate_flag</b>	5	u(1)
if( average_rate_flag ) {		
<b>accurate_statistics_flag</b>	5	u(1)
<b>average_bit_rate</b>	5	u(16)
<b>average_frame_rate</b>	5	u(16)
}		
<b>num_referenced_subseqs</b>	5	ue(v)
for( n = 0; n < num_referenced_subseqs; n++ ) {		
<b>ref_sub_seq_layer_num</b>	5	ue(v)
<b>ref_sub_seq_id</b>	5	ue(v)
<b>ref_sub_seq_direction</b>	5	u(1)
}		
}		

#### D.1.14 Sintaxis del mensaje SEI de congelación de todo el cuadro

full_frame_freeze( payloadSize ) {	<b>C</b>	<b>Descriptor</b>
full_frame_freeze_repetition_period	5	ue(v)
}		

#### D.1.15 Sintaxis del mensaje SEI de liberación de congelación de todo el cuadro

full_frame_freeze_release( payloadSize ) {	<b>C</b>	<b>Descriptor</b>
}		

#### D.1.16 Sintaxis del mensaje SEI de imagen instantánea de cuadro completo

full_frame_snapshot( payloadSize ) {	<b>C</b>	<b>Descriptor</b>
snapshot_id	5	ue(v)
}		

#### D.1.17 Sintaxis del mensaje SEI de comienzo del segmento de refinamiento progresivo

progressive_refinement_segment_start( payloadSize ) {	<b>C</b>	<b>Descriptor</b>
progressive_refinement_id	5	ue(v)
num_refinement_steps_minus1	5	ue(v)
}		

#### D.1.18 Sintaxis del mensaje SEI de fin del segmento de refinamiento progresivo

progressive_refinement_segment_end( payloadSize ) {	<b>C</b>	<b>Descriptor</b>
progressive_refinement_id	5	ue(v)
}		

#### D.1.19 Sintaxis del mensaje SEI de conjunto de grupos de sectores de movimiento limitado

motion_constrained_slice_group_set( payloadSize ) {	<b>C</b>	<b>Descriptor</b>
num_slice_groups_in_set_minus1	5	ue(v)
for( i = 0; i <= num_slice_groups_in_set_minus1; i++)		
slice_group_id[ i ]	5	u(v)
exact_sample_value_match_flag	5	u(1)
pan_scan_rect_flag	5	u(1)
if( pan_scan_rect_flag )		
pan_scan_rect_id	5	ue(v)
}		

## D.1.20 Sintaxis del mensaje SEI reservado

	C	Descriptor
reserved_sei_message( payloadSize ) {		
for( i = 0; i < payloadSize; i++ )		
<b>reserved_sei_message_payload_byte</b>	5	b(8)
}		

## D.2 Semántica de la cabida útil del mensaje SEI

### D.2.1 Semántica del mensaje SEI de periodo de almacenamiento intermedio

Cuando `NalHrdBpPresentFlag` o `VclHrdBpPresentFlag` es igual a 1, se puede asociar un mensaje SEI de periodo de almacenamiento intermedio a cualquier unidad de acceso en el tren de bits, y se asociará obligatoriamente a cada unidad de acceso IDR y a cada unidad de acceso asociada a un mensaje SEI de punto de recuperación.

NOTA – En algunas aplicaciones, puede ser conveniente que haya frecuentemente un mensaje SEI de periodo de almacenamiento intermedio.

El periodo de almacenamiento intermedio se especifica mediante un conjunto de unidades de acceso entre dos ejemplares del mensaje SEI de periodo de almacenamiento intermedio en orden de decodificación.

**seq\_parameter\_set\_id** especifica el conjunto de parámetros secuencia que contiene los atributos HRD de la secuencia. El valor de `seq_parameter_set_id` será igual al valor de `seq_parameter_set_id` en el conjunto de parámetros de imagen referenciados por la imagen codificada primaria asociada al mensaje SEI de periodo de almacenamiento intermedio. El valor de `seq_parameter_set_id` estará entre 0 y 31, inclusive.

**initial\_cpb\_removal\_delay**[ SchedSelIdx ] especifica el retardo de la SchedSelIdx-ésima CPB entre el instante de llegada a la CPB del primer bit de los datos codificados correspondientes a la unidad de acceso asociada al mensaje SEI de periodo de almacenamiento intermedio y el instante de extracción de la CPB de los datos codificados correspondientes a la misma unidad de acceso, para el primer periodo de almacenamiento intermedio después de la inicialización del HRD. El elemento sintáctico tiene una longitud en bits dada por `initial_cpb_removal_delay_length_minus1 + 1`, en unidades de un reloj de 90 kHz. `initial_cpb_removal_delay`[ SchedSelIdx ] será distinto de 0 y no será mayor que  $90000 * ( \text{CpbSize}[\text{SchedSelIdx}] \div \text{BitRate}[\text{SchedSelIdx}] )$ , el equivalente de tiempo del tamaño de la CPB en unidades de reloj de 90 kHz.

**initial\_cpb\_removal\_delay\_offset**[ SchedSelIdx ] se utiliza para SchedSelIdx-ésima CPB junto con `cpb_removal_delay` para especificar el instante de entrega inicial de las unidades de acceso codificadas a la CPB. `initial_cpb_removal_delay_offset`[ SchedSelIdx ] está dado en unidades de un reloj de 90 kHz. El elemento sintáctico `initial_cpb_removal_delay_offset`[ SchedSelIdx ] es un código de longitud constante que viene dada en bits por `initial_cpb_removal_delay_length_minus1 + 1`. Los decodificadores no utilizan este elemento sintáctico, que sólo es necesario para el planificador de entrega (HSS) especificado en el anexo C.

En toda la secuencia de vídeo codificada, la suma de `initial_cpb_removal_delay`[ SchedSelIdx ] y `initial_cpb_removal_delay_offset`[ SchedSelIdx ] debe ser constante para cada valor de SchedSelIdx.

### D.2.2 Semántica del mensaje SEI de temporización de imágenes

Cuando `CpbDpbDelaysPresentFlag` es igual a 1, se asociará un mensaje SEI de temporización de imágenes a cada unidad de acceso en el tren de bits.

**cpb\_removal\_delay** indica cuántos tics de reloj (véase la subcláusula E.2.1) deben pasar desde que se extrae de la CPB la unidad de acceso asociada al mensaje SEI de periodo de almacenamiento

intermedio más reciente hasta que se extraen de la memoria intermedia los datos de la unidad de acceso asociados al mensaje SEI de temporización de imágenes. Este valor se utiliza también para calcular un instante de llegada más prematuro posible de los datos de la unidad de acceso a la CPB para el HSS, conforme al anexo C. El elemento sintáctico es un código de longitud constante cuya longitud en bits está dada por  $\text{cpb\_removal\_delay\_length\_minus1} + 1$ . El  $\text{cpb\_removal\_delay}$  es el residuo de un contador  $2^{(\text{cpb\_removal\_delay\_length\_minus1} + T)}$ .

El valor de  $\text{cpb\_removal\_delay}$  para la primera imagen en el tren de bits será igual a 0.

**dpb\_output\_delay** se utiliza para calcular el tiempo de salida de la imagen de la DPB. Éste indica cuántos tics de reloj deben pasar desde la extracción de una unidad de acceso de la CPB hasta que la imagen decodificada se pueda sacar de la DPB (véase la subcláusula C.2).

NOTA – Una imagen no se extrae de la DPB en su instante de salida si aún está marcada como "utilizada como referencia de corto alcance" o "utilizada como referencia de largo alcance".

NOTA – Para una imagen decodificada sólo se especifica un  $\text{dpb\_output\_delay}$ .

El tamaño del elemento de sintaxis  $\text{dpb\_output\_delay}$  está dado en bits por  $\text{dpb\_output\_delay\_length\_minus1} + 1$ . Si  $\text{max\_dec\_frame\_buffering}$  es igual a 0,  $\text{dpb\_output\_delay}$  ha de ser igual a 0.

El instante de salida calculado a partir de  $\text{dpb\_output\_delay}$  de cualquier imagen que sale de un decodificador conforme con la temporización de salida especificada en la subcláusula C.2 será anterior al instante de salida calculado a partir de  $\text{dpb\_output\_delay}$  de todas las imágenes de cualquier secuencia de vídeo codificado subsiguiente en orden de decodificación.

El instante de salida calculado a partir de  $\text{dpb\_output\_delay}$  del segundo campo, en orden de decodificación, de un par de campos complementarios que no son de referencia será posterior al instante de salida calculado a partir de  $\text{dpb\_output\_delay}$  del primer campo del mismo par de campos complementarios que no son de referencia.

El orden de salida de la imagen establecido por los valores de este elemento sintáctico será el mismo orden establecido por los valores de  $\text{PicOrderCnt}()$  conforme a las subcláusulas C.4.1 a C.4.5, salvo cuando los dos campos de un par de campos de referencia complementarios tienen el mismo valor de  $\text{PicOrderCnt}()$ , en cuyo caso los dos campos tienen distintos instantes de salida.

En el caso de imágenes que no salen por el proceso de "vaciado" descrito en la subcláusula C.4.5 porque están antes, en orden de decodificación, de una imagen IDR cuyo  $\text{no\_output\_of\_prior\_pics\_flag}$  es igual a 1 o que se infiere igual a 1, los tiempos de salida calculados a partir de  $\text{dpb\_output\_delay}$  aumentarán con el aumento del valor de  $\text{PicOrderCnt}()$  en relación con todas las imágenes dentro de la misma secuencia de vídeo codificado subsiguientes a cualquier imagen que tenga una  $\text{memory\_management\_control\_operation}$  igual a 5.

**pic\_struct** indica si una imagen se debe visualizar como cuadro o como uno o más campos, conforme al cuadro D-1. La duplicación de cuadro ( $\text{pic\_struct}$  igual a 7) indica que el cuadro se debe visualizar dos veces consecutivas, y la triplicación de cuadro ( $\text{pic\_struct}$  igual a 8) indica que el cuadro se debe visualizar tres veces consecutivas.

NOTA – La duplicación de cuadro puede facilitar la visualización de, por ejemplo, vídeo de 25p en una pantalla de 50p y vídeo de 29,97p en una pantalla de 59,94p. Combinando la duplicación y triplicación de cuadros en uno de cada dos cuadros se puede facilitar la visualización de vídeo de 23,98p en una pantalla de 59,94p.

**Cuadro D-1 – Interpretación de pic\_struct**

Valor	Visualización de imagen indicada	Restricciones	NumClockTS
0	Cuadro	field_pic_flag debe ser 0	1
1	Campo superior	field_pic_flag debe ser 1, bottom_field_flag debe ser 0	1
2	Campo inferior	field_pic_flag debe ser 1, bottom_field_flag debe ser 1	1
3	Campo superior, campo inferior, en ese orden	field_pic_flag debe ser 0	2
4	Campo inferior, campo superior, en ese orden	field_pic_flag debe ser 0	2
5	Campo superior, campo inferior, campo superior repetido, en ese orden	field_pic_flag debe ser 0	3
6	Campo inferior, campo superior, campo inferior repetido, en ese orden	field_pic_flag debe ser 0	3
7	Duplicación de cuadro	field_pic_flag debe ser 0 fixed_frame_rate_flag debe ser 1	2
8	Triplicación de cuadro	field_pic_flag debe ser 0 fixed_frame_rate_flag debe ser 1	3
9..15	Reservados		

NumClockTS se determina mediante pic\_struct según se indica en el cuadro D-1. Hay hasta NumClockTS conjuntos de información de sello de tiempo de reloj para una imagen, como indica clock\_timestamp\_flag[ i ] de cada conjunto. Los conjuntos de información de sello de tiempo de reloj se aplican al campo o campos o al cuadro o cuadros relacionados con la imagen mediante pic\_struct.

El contenido de los elementos sintácticos sello de tiempo de reloj indican un instante de origen, captura o visualización ideal alternativa. Este instante indicado se calcula así:

$$\text{clockTimestamp} = ( ( \text{hH} * 60 + \text{mM} ) * 60 + \text{sS} ) * \text{time\_scale} + \text{nFrames} * ( \text{num\_units\_in\_tick} * ( 1 + \text{nuit\_field\_based\_flag} ) ) + \text{tOffset}, \quad (\text{D-1})$$

en unidades de tics de un reloj de frecuencia igual a time\_scale Hz, relativo a un punto no especificado en el tiempo en el cual clockTimestamp es igual a 0. Las temporizaciones de orden de salida y de salida de la DPB no se ven afectadas por el valor de clockTimestamp. Cuando dos o más cuadros con pic\_struct igual a 0 son consecutivos en orden de salida y tienen los mismos valores de clockTimestamp, significa que los cuadros representan el mismo contenido y que el último de estos cuadros en orden de salida es la representación preferida.

NOTA – Las indicaciones de tiempo clockTimestamp pueden ayudar a la visualización en dispositivos con velocidades de regeneración distintas a aquellas que están bien coordinadas con los instantes de salida de la DPB.

**clock\_timestamp\_flag[ i ]** igual a 1 indica que inmediatamente después hay varios elementos sintácticos sello de tiempo de reloj clock\_timestamp\_flag[ i ] igual a 0 indica que no hay elementos sintácticos sello de tiempo de reloj asociados. Cuando NumClockTS es mayor que 1 y clock\_timestamp\_flag[ i ] es igual a 1 para más de un valor de i, el valor de clockTimestamp no disminuirá al aumentar el valor de i.

**ct\_type** indica el tipo de barrido (entrelazado o progresivo) del material fuente como sigue:

Los dos campos de un cuadro codificado pueden tener valores diferentes de ct\_type.

Cuando clockTimestamp es igual para dos campos de paridad contraria que son consecutivos en orden de salida, ambos con ct\_type igual a 0 (progresivo) o ct\_type igual a 2 (desconocido), significa que los dos campos provienen del mismo cuadro progresivo original. Dos campos consecutivos en orden de salida tendrán valores distintos de clockTimestamp cuando el valor de ct\_type de cada uno de los campos es 1 (entrelazado).

**Cuadro D-2 – Correspondencia de ct\_type con el barrido de la imagen fuente**

Valor	Barrido de la imagen original
0	Progresivo
1	Entrelazado
2	Desconocido
3	Reservado

**nuit\_field\_based\_flag:** Se utiliza para calcular clockTimestamp, conforme a la ecuación D-1.

**counting\_type:** Especifica el método de supresión de valores de n\_frames conforme al cuadro D-3.

**Cuadro D-3 – Definición de los valores counting\_type**

Valor	Interpretación
0	No se suprimen valores de recuento de n_frames y no se utiliza time_offset
1	No se suprimen valores de recuento de n_frames
2	Se suprimen valores cero particulares del recuento de n_frames
3	Se suprimen valores MaxFPS-1 particulares del recuento de n_frames
4	Se suprimen los dos recuentos más bajos (valores 0 y 1) de n_frames cuando seconds_value es igual a 0 y minutes_value no es un valor entero múltiplo de 10
5	Se suprimen valores de recuento de n_frames particulares no especificados
6	Se suprimen números no especificados de valores de recuento de n_frames no especificados
7..31	Reservados

**full\_timestamp\_flag** igual a 1 indica que después del elemento sintáctico n\_frames vienen seconds\_value, minutes\_value y hours\_value. full\_timestamp\_flag igual a 0 indica que después del elemento sintáctico n\_frames viene seconds\_flag.

**discontinuity\_flag** igual a 0 indica que la diferencia entre el valor actual de clockTimestamp y el valor de clockTimestamp calculado a partir del sello de tiempo de reloj anterior en orden de salida puede interpretarse como la diferencia de tiempo entre los instantes de origen o adquisición de los correspondientes cuadros o campos. discontinuity\_flag igual a 1 indica que la diferencia entre el valor actual de clockTimestamp y el valor de clockTimestamp calculado a partir del sello de tiempo de reloj anterior en orden de salida no se debe interpretar como la diferencia de tiempo entre los instantes de origen o adquisición de los correspondientes cuadros o campos. Cuando discontinuity\_flag es igual a 0, el valor de clockTimestamp será mayor o igual que todos los valores de clockTimestamp de la imagen precedente en el orden de salida de la DPB.

**cnt\_dropped\_flag** especifica que se obvian uno o más valores de n\_frames mediante el método de recuento indicado por counting\_type.

**n\_frames** indica el valor de nFrames utilizado para calcular clockTimestamp. n\_frames será menor que

$$\text{MaxFPS} = \text{Ceil}(\text{time\_scale} \div \text{num\_units\_in\_tick}) \quad (\text{D-2})$$

NOTA – n\_frames es un contador de cuadros. Para las indicaciones de temporización específicas de campo, se debe utilizar time\_offset para indicar un clockTimestamp distinto para cada campo.

Cuando counting\_type es igual a 2 y cnt\_dropped\_flag es igual a 1, n\_frames será igual a 1 y el valor de n\_frames de la imagen previa en orden de salida será distinto de 0 a menos que discontinuity\_flag sea igual a 1.

NOTA – Cuando counting\_type es igual a 2, se puede evitar la necesidad de usar magnitudes cada vez más grandes de tOffset en la ecuación D-1 cuando se utilizan velocidades de cuadro constantes no enteras (por ejemplo, 12,5 cuadros por segundo con time\_scale igual a 25 y num\_units\_in\_tick igual a 2 y nuit\_field\_based\_flag igual a 0) obviando de vez en cuando los n\_frames de valor 0 cuando se hace el recuento (por ejemplo, se hace el recuento de n\_frames de 0 a 12, a continuación se aumenta seconds\_value y se hace el recuento de n\_frames de 1 a 12, a continuación se aumenta seconds\_value y se hace el recuento de n\_frames de 0 a 12, etc.).

Cuando counting\_type es igual a 3 y cnt\_dropped\_flag es igual a 1, n\_frames será igual a 0 y el valor de n\_frames para la imagen anterior en orden de salida será distinto de MaxFPS – 1 a menos que discontinuity\_flag sea igual a 1.

NOTA – Cuando counting\_type es igual a 3, se puede evitar la necesidad de usar magnitudes cada vez más grandes de tOffset en la ecuación D-1 cuando se utilizan velocidades de cuadro constantes no enteras (por ejemplo, 12,5 cuadros por segundo con time\_scale igual a 25 y num\_units\_in\_tick igual a 2 y nuit\_field\_based\_flag igual a 0) obviando de vez en cuando los n\_frames de valor MaxFPS cuando se hace el recuento (por ejemplo, se hace el recuento de n\_frames de 0 a 12, a continuación se aumenta seconds\_value y se hace el recuento de n\_frames de 0 a 11, a continuación se aumenta seconds\_value y se hace el recuento de n\_frames de 0 a 12, etc.).

Cuando counting\_type es igual a 4 y cnt\_dropped\_flag es igual a 1, n\_frames será igual a 2, el valor especificado de sS será cero, el valor especificado de mM no será un múltiplo entero de diez y n\_frames de la imagen previa en orden de salida no será igual a 0 ó 1 a menos que discontinuity\_flag sea igual a 1.

NOTA – Cuando counting\_type es igual a 4, se puede reducir la necesidad de usar magnitudes cada vez más grandes de tOffset en la ecuación D-1 cuando se utilizan velocidades de cuadro constantes no enteras (por ejemplo, 30 000 ÷ 1001 cuadros por segundo con time\_scale igual a 60 000 y num\_units\_in\_tick igual a 1001 y nuit\_field\_based\_flag igual a 1) obviando de vez en cuando los n\_frames de valor MaxFPS cuando se hace el recuento (por ejemplo, se hace el recuento de n\_frames de 0 a 29, a continuación se aumenta seconds\_value y se hace el recuento de n\_frames de 0 a 29, etc., hasta que seconds\_value sea cero y minutes\_value no sea un múltiplo entero de diez, después se hace el recuento de n\_frames de 2 a 29, a continuación se aumenta seconds\_value y se hace el recuento de n\_frames de 0 a 29, etc.). Este método de recuento es muy utilizado en la industria y a menudo se denomina recuento de "supresión de cuadros NTSC".

Cuando counting\_type es igual a 5 ó 6 y cnt\_dropped\_flag es igual a 1, n\_frames será distinto de 1 más el valor de n\_frames de la imagen previa en orden de salida módulo MaxFPS a menos que discontinuity\_flag sea igual a 1.

NOTA – Cuando counting\_type es igual a 5 ó 6, se puede evitar la necesidad de usar magnitudes cada vez más grandes de tOffset en la ecuación D-1 cuando se utilizan velocidades de cuadro constantes no enteras obviando de vez en cuando algunos valores de n\_frames cuando se lleva a cabo el recuento. Los valores específicos de n\_frames que se obvian no están especificados cuando counting\_type es igual a 5 ó 6.

**seconds\_flag** igual a 1 indica que hay seconds\_value y minutes\_flag cuando full\_timestamp\_flag es igual a 0. seconds\_flag igual a 0 indica que no hay seconds\_value y minutes\_flag.



**seconds\_value** indica el valor de sS utilizado para calcular clockTimestamp. El valor de seconds\_value estará entre 0 y 59, inclusive. Cuando no hay seconds\_value, se utilizará el seconds\_value anterior en orden de decodificación utilizado como sS para calcular clockTimestamp.

**minutes\_flag** igual a 1 indica que hay minutes\_value y hours\_flag cuando full\_timestamp\_flag es igual a 0 y seconds\_flag es igual a 1. minutes\_flag igual a 0 indica que no hay minutes\_value y hours\_flag.

**minutes\_value** indica el valor de mM que se utiliza para calcular clockTimestamp. El valor de minutes\_value estará entre 0 y 59, inclusive. Cuando no hay minutes\_value, se utilizará el minutes\_value anterior en orden de decodificación utilizado como mM para calcular clockTimestamp.

**hours\_flag** igual a 1 indica que hay hours\_value cuando full\_timestamp\_flag es igual a 0 y seconds\_flag es igual a 1 y minutes\_flag es igual a 1.

**hours\_value** indica el valor de hH que se utiliza para calcular clockTimestamp. El valor de hours\_value estará entre 0 y 23, inclusive. Cuando no hay hours\_value, se utilizará el hours\_value anterior en orden de decodificación utilizado como hH para calcular clockTimestamp.

**time\_offset** indica el valor de tOffset que se utiliza para calcular clockTimestamp. El número de bits empleado para representar time\_offset será igual a time\_offset\_length. Cuando no haya time\_offset, se utilizará 0 como valor de tOffset para calcular clockTimestamp.

### D.2.3 Semántica del mensaje SEI de rectángulo de selección

Los elementos sintácticos del mensaje SEI de rectángulo de selección (pan-scan) indican las coordenadas de un rectángulo relativas al rectángulo de recorte del conjunto de parámetros secuencia. Las coordenadas del rectángulo se especifican en unidades de un dieciseisavo de la distancia entre muestras de la cuadrícula de muestreo luma.

**pan\_scan\_rect\_id** contiene un número de identificación que se puede utilizar para identificar la finalidad del rectángulo de barrido panorámico (por ejemplo, para indicar una zona que se ha de mostrar en una determinada pantalla o una zona que contiene un determinado actor en la escena). El valor de pan\_scan\_rect\_id estará entre 0 y  $2^{32}-1$ , inclusive.

La utilización de los valores de pan\_scan\_rect\_id de 0 a 255 y de 512 a  $2^{31}-1$  depende de la aplicación. Los valores de pan\_scan\_rect\_id de 256 a 511 y de  $2^{31}$  a  $2^{32}-1$  están reservados para que el UIT-T | ISO/CEI los utilice en el futuro. Los decodificadores que detecten un valor de pan\_scan\_rect\_id en la gama de 256 a 511 o en la gama de  $2^{31}$  a  $2^{32}-1$  no lo tendrán en cuenta (se extrae del tren de bits y se descarta).

**pan\_scan\_rect\_cancel\_flag** igual a 1 indica que el mensaje SEI cancela un mensaje SEI de rectángulo de selección anterior. pan\_scan\_rect\_cancel\_flag igual a 0 indica que el mensaje SEI no cancela al mensaje SEI de rectángulo de selección anterior y que la información de ese rectángulo de selección viene a continuación.

**pan\_scan\_cnt\_minus1** indica el número de rectángulos de selección incluidos en el mensaje SEI. El valor de pan\_scan\_cnt\_minus1 estará entre 0 y 2, inclusive. pan\_scan\_cnt\_minus1 igual a 0 indica que sólo hay un rectángulo de selección, el cual se aplica a todos los campos de la imagen decodificada. pan\_scan\_cnt\_minus1 debe ser igual a 0 cuando la imagen considerada es un campo. pan\_scan\_cnt\_minus1 igual a 1 indica que hay dos rectángulos de selección: el primero se aplica al primer campo de la imagen en orden de salida y el segundo se aplica al segundo campo de la imagen en orden de salida. pan\_scan\_cnt\_minus1 igual a 2 indica que hay tres rectángulos de selección: el primero se aplica al primer campo de la imagen en orden de salida, el segundo se aplica al segundo campo de la imagen en orden de salida y el tercero se aplica de nuevo al primer campo para generar un tercer campo en orden de salida.

**pan\_scan\_rect\_left\_offset[ i ]**, **pan\_scan\_rect\_right\_offset[ i ]**, **pan\_scan\_rect\_top\_offset[ i ]**, y **pan\_scan\_rect\_bottom\_offset[ i ]**, indican la ubicación del rectángulo de selección, mediante números enteros con signo en unidades de un dieciseisavo de la distancia entre dos muestras de la cuadrícula de muestreo luma. Los valores de cada uno de estos cuatro elementos sintácticos estarán entre  $-2^{31}$  y  $2^{31} - 1$ , inclusive.

Las coordenadas del rectángulo de selección, en unidades de un dieciseisavo de la distancia entre dos muestras de la cuadrícula de muestreo de cuadros luma, son las siguientes:

- Si **frame\_mbs\_only\_flag** es igual a 1, las coordenadas horizontales de cuadro luma son  $32 * \text{frame\_crop\_left\_offset} + \text{pan\_scan\_rect\_left\_offset}[ i ]$  y  $32 * ( 8 * \text{PicWidthInMbs} - \text{frame\_crop\_right\_offset} ) + \text{pan\_scan\_rect\_right\_offset}[ i ] - 1$ ; y las coordenadas verticales son  $32 * \text{frame\_crop\_top\_offset} + \text{pan\_scan\_rect\_top\_offset}[ i ]$  y  $32 * ( 8 * \text{PicHeightInMbs} - \text{frame\_crop\_bottom\_offset} ) + \text{pan\_scan\_rect\_bottom\_offset}[ i ] - 1$ , inclusive. En este caso, el valor de  $32 * \text{frame\_crop\_left\_offset} + \text{pan\_scan\_rect\_left\_offset}[ i ]$  será menor o igual que  $32 * ( 8 * \text{PicWidthInMbs} - \text{frame\_crop\_right\_offset} ) + \text{pan\_scan\_rect\_right\_offset}[ i ] - 1$ ; y el valor de  $32 * \text{frame\_crop\_top\_offset} + \text{pan\_scan\_rect\_top\_offset}[ i ]$  será menor o igual que  $32 * ( 8 * \text{PicHeightInMbs} - \text{frame\_crop\_bottom\_offset} ) + \text{pan\_scan\_rect\_bottom\_offset}[ i ] - 1$ .
- De lo contrario (**frame\_mbs\_only\_flag** es igual a 0), las coordenadas horizontales de cuadro luma son  $32 * \text{frame\_crop\_left\_offset} + \text{pan\_scan\_rect\_left\_offset}[ i ]$  y  $32 * ( 8 * \text{PicWidthInMbs} - \text{frame\_crop\_right\_offset} ) + \text{pan\_scan\_rect\_right\_offset}[ i ] - 1$ ; y las coordenadas verticales son  $64 * \text{frame\_crop\_top\_offset} + \text{pan\_scan\_rect\_top\_offset}[ i ]$  y  $64 * ( 4 * \text{PicHeightInMbs} - \text{frame\_crop\_bottom\_offset} ) + \text{pan\_scan\_rect\_bottom\_offset}[ i ] - 1$ , inclusive. En este caso, el valor de  $32 * \text{frame\_crop\_left\_offset} + \text{pan\_scan\_rect\_left\_offset}[ i ]$  será menor o igual que  $32 * ( 8 * \text{PicWidthInMbs} - \text{frame\_crop\_right\_offset} ) + \text{pan\_scan\_rect\_right\_offset}[ i ] - 1$ ; y el valor de  $64 * \text{frame\_crop\_top\_offset} + \text{pan\_scan\_rect\_top\_offset}[ i ]$  será menor o igual que  $64 * ( 4 * \text{PicHeightInMbs} - \text{frame\_crop\_bottom\_offset} ) + \text{pan\_scan\_rect\_bottom\_offset}[ i ] - 1$ .

Cuando la zona rectangular seleccionada incluye muestras que están fuera del rectángulo de recorte, la región fuera de este rectángulo se puede rellenar con contenido sintetizado (por ejemplo contenido de vídeo negro o de vídeo gris neutro) para su visualización.

**pan\_scan\_rect\_repetition\_period** indica si habrá en el tren de bits otro mensaje SEI de rectángulo de selección con el mismo valor de **pan\_scan\_rect\_id** e indica en qué intervalo de número de orden de imágenes estará. El valor de **pan\_scan\_rect\_repetition\_period** estará entre 0 y 16 384, inclusive. Cuando **pan\_scan\_cnt\_minus1** es mayor que 0, **pan\_scan\_rect\_repetition\_period** no será mayor que 1.

**pan\_scan\_rect\_repetition\_period** igual a 0 indica que la información del rectángulo de selección se aplica únicamente a la imagen decodificada actual.

**pan\_scan\_rect\_repetition\_period** igual a 1 indica que la información del rectángulo de selección se sigue aplicando en orden de salida hasta que se cumpla alguna de las siguientes condiciones.

- Comienza una nueva secuencia de vídeo codificado.
- Sale una imagen en una unidad de acceso que contiene un mensaje SEI de rectángulo de selección con el mismo valor de **pan\_scan\_rect\_id**, y cuyo **PicOrderCnt( )** es mayor que **PicOrderCnt( CurrPic )**.

**pan\_scan\_rect\_repetition\_period** igual a 0 ó 1 indica que puede haber o no otro mensaje SEI de rectángulo de selección con el mismo valor de **pan\_scan\_rect\_id**.

pan\_scan\_rect\_repetition\_period mayor que 1 indica que la información del rectángulo de selección se sigue aplicando hasta que se cumpla alguna de las siguientes condiciones.

- Comienza una nueva secuencia de vídeo codificado.
- Sale una imagen en una unidad de acceso que incluye un mensaje SEI de rectángulo de selección con el mismo valor de pan\_scan\_rect\_id y cuyo PicOrderCnt() es mayor que PicOrderCnt( CurrPic ) + pan\_scan\_rect\_repetition\_period.

pan\_scan\_rect\_repetition\_period mayor que 1 indica que habrá otro mensaje SEI de rectángulo de selección con el mismo valor de pan\_scan\_rect\_id de una imagen en una unidad de acceso que se sale y cuyo PicOrderCnt() es menor o igual que PicOrderCnt( CurrPic ) + pan\_scan\_rect\_repetition\_period, a menos que comience una nueva secuencia de vídeo codificado sin que salga esa imagen.

#### **D.2.4 Semántica del mensaje SEI de cabida útil de relleno**

Este mensaje incluye una serie de bytes payloadSize de valor 0xFF, que se pueden descartar.

ff\_byte será un byte de valor 0xFF.

#### **D.2.5 Semántica del mensaje SEI de datos de usuario registrados conforme a la Rec. UIT-T T.35**

Este mensaje contiene datos de usuario registrados conforme a la Rec. UIT-T T.35, cuyo contenido no se especifica en esta Recomendación | Norma Internacional.

itu\_t\_t35\_country\_code es un byte cuyo valor será un indicativo de país de los que figuran en el anexo A/T.35.

itu\_t\_t35\_country\_code\_extension\_byte es un byte cuyo valor será un indicativo de país de los que se especifican en el anexo B/T.35.

itu\_t\_t35\_payload\_byte es un byte que contendrá datos registrados conforme a la Rec. UIT-T T.35.

El código de proveedor de terminal y el código de terminal definido por el proveedor de la Rec. UIT-T T.35 se incluirán en los primeros bytes de itu\_t\_t35\_payload\_byte, en el formato especificado por la Administración que expidió el código de proveedor de terminal. Los datos itu\_t\_t35\_payload\_byte restantes contendrán sintaxis y semántica especificada por la entidad identificada por el indicativo de país y el código de proveedor terminal de la Rec. UIT-T T.35.

#### **D.2.6 Semántica del mensaje SEI de datos de usuario no registrados**

Este mensaje incluye datos de usuario no registrados que se identifican mediante un UUID, cuyo contenido no se especifica en esta Recomendación | Norma Internacional.

uuid\_iso\_iec\_11578 contendrá un UUID de acuerdo con los procedimientos del anexo A de ISO/CEI 11578:1996.

user\_data\_payload\_byte es un byte que contendrá datos de sintaxis y semántica especificadas por el generador de UUID.

#### **D.2.7 Semántica del mensaje SEI de punto de recuperación**

El mensaje SEI de punto de recuperación ayuda al decodificador a determinar cuándo el proceso de decodificación produce imágenes aceptables para su visualización, después de que el codificador haya iniciado un acceso aleatorio o después de que el codificador indique una interrupción de enlace en la secuencia. Cuando el proceso de decodificación comienza con la unidad de acceso en orden de decodificación correspondiente al mensaje SEI de punto de recuperación, el contenido de todas las imágenes decodificadas en el punto de recuperación, o imágenes posteriores en orden de salida, especificado en este mensaje SEI se indica que es correcto o aproximadamente correcto. Las

imágenes decodificadas generadas mediante acceso aleatorio a la imagen correspondiente al mensaje SEI de punto de recuperación, o antes de ella, no tienen por qué tener contenido correcto antes del punto de recuperación indicado, y el funcionamiento del proceso de decodificación que comienza en la imagen correspondiente al mensaje SEI de punto de recuperación puede contener referencias a imágenes que no están en la memoria intermedia de imágenes decodificadas.

Además, si se utiliza `broken_link_flag`, el mensaje SEI de punto de recuperación puede indicar al decodificador la ubicación de algunas imágenes en el tren de bits que pueden provocar grandes distorsiones visuales cuando se visualicen, aun cuando el proceso de decodificación haya comenzado en la ubicación de una unidad de acceso IDR anterior en orden de decodificación.

NOTA – Los codificadores pueden emplear `broken_link_flag` para indicar la ubicación de un punto después del cual el proceso de decodificación de algunas imágenes puede hacer referencia a imágenes que, aunque se utilicen en el proceso de decodificación, no se utilizaron como referencia cuando se codificó el tren de bits originalmente (por ejemplo, por haber realizado una unión al generar el tren de bits).

El punto de recuperación se especifica como el número de incrementos de `frame_num` posteriores al `frame_num` de la unidad de acceso actual en la posición del mensaje SEI.

NOTA – Cuando el tren de bits contiene información HRD, se debe asociar un mensaje SEI de periodo de almacenamiento intermedio a la unidad de acceso correspondiente al mensaje SEI de punto de recuperación a fin de establecer la inicialización del modelo de memoria intermedia HRD después de un acceso aleatorio.

**recovery\_frame\_cnt** indica el punto de recuperación de las imágenes de salida en orden de salida. El contenido de todas las imágenes decodificadas en orden de salida se indicará que es correcto o aproximadamente correcto a partir de la posición en orden de salida de la imagen de referencia cuyo `frame_num` es igual al `frame_num` de las unidades NAL VCL de la unidad de acceso actual más `recovery_frame_cnt` módulo aritmético `MaxFrameNum`. `recovery_frame_cnt` estará entre 0 y `MaxFrameNum` – 1, inclusive.

**exact\_match\_flag** indica si las imágenes decodificadas en el punto de recuperación especificado, y las subsiguientes en orden de salida, resultantes de iniciar el proceso de decodificación en la unidad de acceso correspondiente al mensaje SEI de punto de recuperación serán exactamente iguales a las imágenes que se hubieran generado al iniciar el proceso de decodificación en la posición de una unidad de acceso IDR anterior en el tren de unidades NAL. El valor 0 indica que la correspondencia no tiene que ser exacta y el valor 1 indica que sí debe ser exacta.

Cuando la decodificación se inicia en la posición del mensaje SEI de punto de recuperación, todas las referencias a imágenes de referencia que no están disponibles se inferirán como referencias a imágenes que sólo contienen macrobloques codificados utilizando modos de predicción Intra de macrobloques y con valores de muestras dados por muestras Y iguales a 128, muestras Cb iguales a 128, y muestras Cr iguales a 128 (gris de nivel medio) para determinar la conformidad del valor de `exact_match_flag`.

NOTA – Al realizar un acceso aleatorio, los decodificadores deben inferir todas las referencias a imágenes de referencia no disponibles como referencias a imágenes que sólo contienen macrobloques Intra y con los valores de las muestras Y igual a 128, Cb igual a 128, y Cr igual a 128 (gris de nivel medio), independientemente del valor de `exact_match_flag`.

Cuando `exact_match_flag` es igual a 0, la calidad de la aproximación en el punto de recuperación la selecciona el proceso de codificación y no se especifica en esta Recomendación | Norma Internacional.

**broken\_link\_flag** indica la presencia o ausencia de un enlace interrumpido en el tren de unidades NAL en la ubicación del mensaje SEI de punto de recuperación y se asigna la siguiente semántica adicional:

- Si `broken_link_flag` es igual a 1, es posible que las imágenes producidas al iniciar el proceso de decodificación en la ubicación de una unidad de acceso IDR anterior contengan distorsiones visuales indeseables de grado tal que las imágenes decodificadas en la unidad de acceso correspondiente al mensaje SEI de punto de recuperación, y en las subsiguientes en orden de decodificación, no deberían visualizarse hasta el punto de recuperación especificado en el orden de salida.
- De lo contrario (`broken_link_flag` es igual a 0), no se da ninguna indicación con relación a la posible aparición de distorsiones visuales.

Independientemente del valor de `broken_link_flag`, el contenido de las imágenes posteriores al punto de recuperación indicado en orden de salida se indica que es correcto o aproximadamente correcto.

NOTA – Cuando hay un mensaje SEI de información de subsecuencia junto con un mensaje SEI de punto de recuperación en el cual `broken_link_flag` es igual a 1 y donde `sub_seq_layer_num` es igual a 0, `sub_seq_id` debe ser diferente del último `sub_seq_id` del `sub_seq_layer_num` igual a 0 que se decodificó antes de la posición del mensaje SEI de punto de recuperación. Cuando `broken_link_flag` es igual a 0, el `sub_seq_id` en la capa de subsecuencia 0 no debe modificarse.

`changing_slice_group_idc` igual a 0 indica que el contenido de las imágenes decodificadas es correcto o aproximadamente correcto en el punto de recuperación, y después del mismo, en orden de salida cuando todos los macrobloques de las imágenes codificadas principales se decodifican dentro del periodo de cambio de grupo de sector, es decir, el periodo entre la unidad de acceso correspondiente al mensaje SEI de punto de recuperación (inclusive) y el punto de recuperación especificado (exclusive) en orden de decodificación. `changing_slice_group_idc` debe ser igual a 0 cuando `num_slice_groups_minus1` es igual a 0 en cualquier imagen codificada principal durante el periodo de cambio de grupo de sector.

Cuando `changing_slice_group_idc` es igual a 1 ó 2, `num_slice_groups_minus1` debe ser igual a 1 y se aplicará el mapeado de macrobloques en grupos de sectores de tipo 3, 4 ó 5 a cada imagen codificada primaria durante el periodo de cambio de grupo de sectores.

`changing_slice_group_idc` igual a 1 indica que durante el periodo de cambio del grupo de sectores no se utilizan valores de muestra que no estén en los macrobloques decodificados abarcados por el grupo de sectores 0 para la predicción inter de cualquier macrobloque dentro del grupo de sectores 0. Además, `changing_slice_group_idc` igual a 1 indica que cuando se decodifican todos los macrobloques del grupo de sectores 0 dentro del periodo de cambio de grupo de sectores, el contenido de las imágenes decodificadas será correcto o aproximadamente correcto en el punto de recuperación especificado y después del mismo en orden de salida independientemente de si se decodificó cualquier macrobloque en el grupo de sectores 1 durante el periodo de cambio de grupo de sectores.

`changing_slice_group_idc` igual a 2 indica que durante el periodo de cambio del grupo de sectores no se utilizan valores de muestra que no estén en los macrobloques decodificados abarcados por el grupo de sectores 1 para la predicción inter de cualquier macrobloque dentro del grupo de sectores 1. Además, `changing_slice_group_idc` igual a 2 indica que cuando se decodifican todos los macrobloques del grupo de sectores 1 dentro del periodo de cambio de grupo de sectores, el contenido de las imágenes decodificadas será correcto o aproximadamente correcto en el punto de recuperación especificado y después del mismo en orden de salida independientemente de si se decodificó cualquier macrobloque del grupo de sectores 0 durante el periodo de cambio de grupo de sectores.

El valor de `changing_slice_group_idc` estará entre 0 y 2, inclusive.

## D.2.8 Semántica del mensaje SEI de repetición de marcado de imágenes de referencia decodificadas

El mensaje SEI de repetición de marcado de imágenes de referencia decodificadas se utiliza para repetir la estructura sintáctica marcado de imágenes de referencia decodificadas que había en la cabecera del sector de una imagen anterior en la secuencia en orden de decodificación.

**original\_idr\_flag** será igual a 1 cuando la estructura sintáctica marcado de imágenes de referencia decodificadas apareció originalmente en una imagen IDR. **original\_idr\_flag** será 0 cuando la estructura sintáctica marcado de imágenes de referencia decodificadas no apareció originalmente en una imagen IDR.

**original\_frame\_num** será igual a **frame\_num** de la imagen en la que apareció originalmente la estructura sintáctica marcado de imágenes de referencia decodificadas que se repite. La imagen a la que apunta **original\_frame\_num** es la imagen codificada anterior que tiene el valor especificado de **frame\_num**. El valor de **original\_frame\_num** utilizado para hacer referencia a una imagen con **memory\_management\_control\_operation** igual a 5 debe ser 0.

**original\_field\_pic\_flag** será igual a **field\_pic\_flag** de la imagen en la que apareció originalmente la estructura sintáctica marcado de imágenes de referencia decodificadas repetida.

**original\_bottom\_field\_flag** será igual a **bottom\_field\_flag** de la imagen en la que apareció originalmente la estructura sintáctica marcado de imágenes de referencia decodificadas repetida.

**dec\_ref\_pic\_marking( )** contendrá una copia de la estructura sintáctica marcado de imágenes de referencia decodificadas de la imagen cuyo **frame\_num** era **original\_frame\_num**. El **nal\_unit\_type** utilizado para especificar la estructura sintáctica **dec\_ref\_pic\_marking( )** repetida será el **nal\_unit\_type** de la cabecera o cabeceras de sector de la imagen cuyo **frame\_num** era **original\_frame\_num** (es decir, el **nal\_unit\_type** que se utiliza conforme a la subcláusula 7.3.3.3 se considerará igual a 5 cuando **original\_idr\_flag** es igual a 1 y se considerará distinto de 5 cuando **original\_idr\_flag** es igual a 0).

## D.2.9 Semántica del mensaje SEI de imagen de reserva

Este mensaje SEI indica que algunas unidades de mapeado de grupos de sectores, denominadas unidades de mapeado de grupos de sectores de reserva, en una o varias imágenes de referencia decodificadas se asemejan a las unidades de mapeado de grupos de sectores cubricados en una determinada imagen decodificada que se denomina imagen objetivo. Las unidades de mapeado de grupos de sectores de reserva se puede utilizar para sustituir en la imagen objetivo unidades de mapeado de grupos de sectores decodificadas incorrectamente y con la misma ubicación. Una imagen decodificada que contiene unidades de mapeado de grupos de sectores de reserva se denomina imagen de reserva.

El valor de **frame\_mbs\_only\_flag** en todas las imágenes de reserva indicadas en un mensaje SEI de imagen de reserva será igual al valor de **frame\_mbs\_only\_flag** de la imagen objetivo indicada en el mismo mensaje SEI. Se aplican las siguientes restricciones a las imágenes de reserva en el mensaje SEI.

- si la imagen objetivo es un campo decodificado, todas las imágenes de reserva indicadas en el mismo mensaje SEI serán campos decodificados;
- de lo contrario (la imagen objetivo es un cuadro decodificado), todas las imágenes de reserva indicadas en el mismo mensaje SEI serán cuadros decodificados.

Los valores de **pic\_width\_in\_mbs\_minus1** y **pic\_height\_in\_map\_units\_minus1** en todas las imágenes de reserva indicadas en un mensaje SEI de imagen de reserva serán iguales a los valores de **pic\_width\_in\_mbs\_minus1** y **pic\_height\_in\_map\_units\_minus1**, respectivamente, de la imagen objetivo indicada en el mismo mensaje SEI. La imagen asociada (conforme a la

subcláusula 7.4.1.2.3) a este mensaje aparecerá después de la imagen objetivo, en orden de decodificación.

**target\_frame\_num** indica el `frame_num` de la imagen objetivo.

**spare\_field\_flag** igual a 0 indica que la imagen objetivo y las imágenes de reserva son cuadros decodificados. **spare\_field\_flag** igual a 1 indica que la imagen objetivo y las imágenes de reserva son campos decodificados.

**target\_bottom\_field\_flag** igual a 0 indica que la imagen objetivo es un campo superior. **target\_bottom\_field\_flag** igual a 1 indica que la imagen objetivo es un campo inferior.

Una imagen objetivo es una imagen de referencia decodificada cuya correspondiente imagen codificada primaria antecede a la imagen actual, en orden de decodificación, y en la cual los valores de `frame_num`, `field_pic_flag` (si está incluida) y `bottom_field_flag` (si los hubiere) son iguales a `target_frame_num`, `spare_field_flag` y `target_bottom_field_flag`, respectivamente.

**num\_spare\_pics\_minus1** indica el número de imágenes de reserva de la imagen objetivo especificada. El número de imágenes de reserva es igual a `num_spare_pics_minus1 + 1`. El valor de `num_spare_pics_minus1` estará entre 0 y 15, inclusive.

**delta\_spare\_frame\_num[ i ]** se utiliza para identificar la imagen de reserva que contiene el *i*-ésimo conjunto de unidades de mapeado de grupos de sectores de reserva, en lo sucesivo denominada *i*-ésima imagen de reserva, como se describe más adelante. El valor de `delta_spare_frame_num[ i ]` estará entre 0 a `MaxFrameNum - 1 - !spare_field_flag`, inclusive.

El `frame_num` de la *i*-ésima imagen de reserva, `spareFrameNum[ i ]`, para todos los valores de *i* de 0 a `num_spare_pics_minus1`, inclusive se calcula del modo siguiente:

```
candidateSpareFrameNum = target_frame_num - !spare_field_flag
for ( i = 0; i <= num_spare_pics_minus1; i++ ) {
    if( candidateSpareFrameNum < 0 )
        candidateSpareFrameNum = MaxFrameNum - 1
    spareFrameNum[ i ] = candidateSpareFrameNum - delta_spare_frame_num[ i ]
    if( spareFrameNum[ i ] < 0 )
        spareFrameNum[ i ] = MaxFrameNum + spareFrameNum[ i ]
    candidateSpareFrameNum = spareFrameNum[ i ] - !spare_field_flag
}
```

(D-3)

**spare\_bottom\_field\_flag[ i ]** igual a 0 indica que la *i*-ésima imagen de reserva es un campo superior. **spare\_bottom\_field\_flag[ i ]** igual a 1 indica que la *i*-ésima imagen de reserva es un campo inferior.

La 0-ésima imagen de reserva es una imagen de referencia decodificada cuya correspondiente imagen codificada primaria precede a la imagen objetivo, en orden de decodificación, y en la cual los valores de `frame_num`, `field_pic_flag` (si los hubiere) y `bottom_field_flag` (si lo hubiere) son iguales a `spareFrameNum[ 0 ]`, `spare_field_flag` y `spare_bottom_field_flag[ 0 ]`, respectivamente. La *i*-ésima imagen de reserva es una imagen de referencia decodificada cuya correspondiente imagen codificada primaria antecede a la (*i* - 1)-ésima imagen de reserva, en orden de decodificación, y en la cual los valores de `frame_num`, `field_pic_flag` (si los hubiere) y `bottom_field_flag` (si lo hubiere) son iguales a `spareFrameNum[ i ]`, `spare_field_flag` y `spare_bottom_field_flag[ i ]`, respectivamente.

**spare\_area\_idc[ i ]** indica el método utilizado para identificar las unidades de mapeado de grupos de sectores de reserva en la *i*-ésima imagen de reserva. `spare_area_idc[ i ]` estará entre 0 y 2, inclusive. `spare_area_idc[ i ]` igual a 0 indica que todas las unidades de mapeado de grupos de sectores en la *i*-ésima imagen de reserva son unidades de reserva. `spare_area_idc[ i ]` igual a 1 indica que se utiliza el valor del elemento sintáctico `spare_unit_flag[ i ][ j ]` para identificar las unidades de mapeado de grupos de sectores de reserva. `spare_area_idc[ i ]` igual a 2 indica que se utiliza el elemento sintáctico `zero_run_length[ i ][ j ]` para deducir los valores de `spareUnitFlagInBoxOutOrder[ i ][ j ]`, como se describe más adelante.

**spare\_unit\_flag[ i ][ j ]** igual a 0 indica que la j-ésima unidad de mapeado de grupos de sectores en orden de barrido por filas en la i-ésima imagen de reserva es una unidad de reserva. **spare\_unit\_flag[ i ][ j ]** igual a 1 indica que la j-ésima unidad de mapeado de grupo de sectores en el orden de barrido por filas en la i-ésima imagen de reserva no es una unidad de reserva.

**zero\_run\_length[ i ][ j ]** se utiliza para calcular los valores de **UnitFlagInBoxOutOrder[ i ][ j ]** cuando **spare\_area\_idc[ i ]** es igual a 2. En este caso, las unidades de mapeado de grupos de sectores de reserva identificadas en **spareUnitFlagInBoxOutOrder[ i ][ j ]** aparecen en el orden de barrido cuadrangular centrífugo dextrógiro, conforme a la subcláusula 8.2.2.4, para cada imagen de reserva. **spareUnitFlagInBoxOutOrder[ i ][ j ]** igual a 0 indica que la j-ésima unidad de mapeado de grupos de sectores en orden de barrido cuadrangular centrífugo dextrógiro en la i-ésima imagen de reserva es una unidad de reserva. **spareUnitFlagInBoxOutOrder[ i ][ j ]** igual a 1 indica que la j-ésima unidad de mapeado de grupos de sectores en orden de barrido cuadrangular centrífugo dextrógiro en la i-ésima imagen de reserva no es una unidad de reserva.

Cuando **spare\_area\_idc[ 0 ]** es igual a 2, **spareUnitFlagInBoxOutOrder[ 0 ][ j ]** se calcula del modo siguiente:

```
for( j = 0, loop = 0; j < PicSizeInMapUnits; loop++ ) {
    for( k = 0; k < zero_run_length[ 0 ][ loop ]; k++ )
        spareUnitFlagInBoxOutOrder[ 0 ][ j++ ] = 0
        spareUnitFlagInBoxOutOrder[ 0 ][ j++ ] = 1
}
```

(D-4)

Cuando **spare\_area\_idc[ i ]** es igual a 2 y el valor de **i** es mayor que 0, **spareUnitFlagInBoxOutOrder[ i ][ j ]** se calcula del modo siguiente:

```
for( j = 0, loop = 0; j < PicSizeInMapUnits; loop++ ) {
    for( k = 0; k < zero_run_length[ i ][ loop ]; k++ )
        spareUnitFlagInBoxOutOrder[ i ][ j ] = spareUnitFlagInBoxOutOrder[ i - 1 ][ j++ ]
        spareUnitFlagInBoxOutOrder[ i ][ j ] = !spareUnitFlagInBoxOutOrder[ i - 1 ][ j++ ]
}
```

(D-5)

## D.2.10 Semántica del mensaje SEI de información de escena

A continuación se describe una escena y una transición de escena mediante un conjunto de imágenes consecutivas en orden de salida.

NOTA – Por lo general, las imágenes decodificadas dentro de una escena tienen contenido similar. El mensaje SEI de información de escena se utiliza para etiquetar imágenes con identificadores de escena e indicar cambios de escena. El mensaje indica cómo se crearon las imágenes fuente de las imágenes etiquetadas. El decodificador puede utilizar esa información para seleccionar un algoritmo apropiado que permita ocultar los errores debidos a la transmisión. Por ejemplo, puede utilizarse un algoritmo específico para ocultar errores debidos a la transmisión que afectan a imágenes que pertenecen a una transición gradual de la escena. Además, cada aplicación define su forma de utilizar el mensaje SEI de información de escena, por ejemplo, para indexar las escenas de una secuencia codificada.

Un mensaje SEI de información de escena etiqueta a todas las imágenes, en orden de decodificación, desde la imagen codificada primaria a la cual está asociado el mensaje SEI (inclusive), conforme a la subcláusula 7.4.1.2.3, hasta la imagen codificada primaria a la cual se asocia el siguiente mensaje SEI de información de escena (si lo hubiere) en orden de decodificación (exclusive) o (de lo contrario) hasta la última unidad de acceso en el tren de bits (inclusive). En este documento esas imágenes se denominan imágenes objetivo.

**scene\_info\_present\_flag** igual a 0 indica que la escena o transición de escena a la que pertenecen las imágenes objetivo no está especificada. **scene\_info\_present\_flag** igual a 1 indica que las imágenes objetivo pertenecen a la misma escena o transición de escena.

**scene\_id** identifica la escena a la que pertenecen las imágenes objetivo. Cuando el valor de **scene\_transition\_type** de las imágenes objetivo es menor que 4, y la imagen anterior en orden de



salida está marcada con un valor `scene_transition_type` menor que 4, y el valor de `scene_id` es el mismo que el valor de `scene_id` de la imagen anterior en orden de salida, significa que el codificador considera que la escena fuente de las imágenes objetivo y la escena fuente de la imagen anterior (en orden de salida) son la misma escena. Cuando el valor de `scene_transition_type` de las imágenes objetivo es mayor que 3, y la imagen anterior en el orden de salida está marcada con un valor `scene_transition_type` menor que 4, y el valor de `scene_id` es el mismo que el valor de `scene_id` de la imagen anterior en orden de salida, significa que el codificador considera que una de las escenas fuente de las imágenes objetivo y la escena fuente de la imagen anterior (en orden de salida) son la misma escena. Cuando el valor de `scene_id` es distinto del valor de `scene_id` de la imagen anterior en el orden de salida, significa que el codificador considera que las imágenes objetivo y la imagen anterior (en el orden de salida) pertenecen a diferentes escenas fuente.

El valor de `scene_id` estará entre 0 y  $2^{32}-1$ , inclusive. La utilización de valores de `scene_id` entre 0 y 255, inclusive, y entre 512 y  $2^{31}-1$ , inclusive, depende de la aplicación. Los valores de `scene_id` entre 256 y 511, inclusive, y entre  $2^{31}$  y  $2^{32}-1$ , inclusive, están reservados para que el UIT-T | ISO/CEI los utilice en el futuro. Los decodificadores que detecten un valor de `scene_id` entre 256 y 511, inclusive, o entre  $2^{31}$  y  $2^{32}-1$ , inclusive, no lo tendrán en cuenta (se extrae del tren de bits y se descarta).

`scene_transition_type` especifica en qué tipo de transición de escena (si la hubiere) están incluidas las imágenes objetivo. Los valores válidos de `scene_transition_type` se especifican en el cuadro D-4.

**Cuadro D-4 – Valores de `scene_transition_type`**

Valor	Descripción
0	Sin transición
1	Color desvanecido a negro
2	Color desvanecido desde negro
3	Transición no especificada desde o hacia color constante
4	Color fundido
5	Transición visual
6	Mezcla no especificada de dos escenas

Cuando `scene_transition_type` es mayor que 3, las imágenes objetivo incluyen contenido de la escena etiquetada mediante su `scene_id` y de la siguiente escena, en orden de salida, que está etiquetada mediante `second_scene_id` (véase más adelante). El término "la escena actual" se utiliza para indicar la escena etiquetada mediante `scene_id`. El término "la siguiente escena" se utiliza para indicar la escena etiquetada mediante `second_scene_id`. No es obligatorio que las siguientes imágenes en orden de salida se etiqueten con `scene_id` igual a `second_scene_id` del mensaje SEI actual.

Los tipos de transición de escena se especifican del modo siguiente:

"Sin transición" indica que las imágenes objetivo forman parte de una transición gradual de escena.

NOTA – Cuando dos imágenes consecutivas en orden de salida tienen `scene_transition_type` igual a 0 y distintos valores de `scene_id`, se produce un corte de escena entre las dos imágenes.

"Fundado en negro" indica que las imágenes objetivo forman parte de una secuencia de imágenes, en orden de salida, que se utilizan en una transición de escena de fundido en negro, es decir, las muestras luma de la escena tienden a cero gradualmente y las muestras croma se tienden a 128 gradualmente.

NOTA – Cuando dos imágenes están etiquetadas como pertenecientes a la misma transición de escena y su `scene_transition_type` es "fundido en negro", la última en orden de salida es más oscura que la anterior.

"Fundido desde negro" indica que las imágenes objetivo forman parte de una secuencia de imágenes en orden de salida que se utilizan en una transición de escena de fundido desde negro, es decir el valor de las muestras luma de la escena aumenta gradualmente de cero y las muestras croma de la escena pueden divergir gradualmente de 128.

NOTA – Cuando dos imágenes están etiquetadas como pertenecientes a la misma transición de escena y su `scene_transition_type` es "fundido desde negro", la última en orden de salida es más clara que la anterior.

"Encadenado" indica que los valores de las muestras de cada imagen objetivo (antes de la codificación) se generaron calculando una suma de valores de muestras ponderadas de una imagen de la escena actual y de una imagen de la siguiente escena. El valor ponderado de la escena actual disminuye gradualmente de su nivel a cero, mientras que el valor ponderado de la siguiente escena aumenta gradualmente de cero a su nivel. Cuando dos imágenes están etiquetadas como pertenecientes a la misma transición de escena y su `scene_transition_type` es "encadenado", en la escena actual el valor ponderado de la imagen posterior en orden de salida es menor que el valor ponderado de la primera de las dos, en la siguiente escena el valor ponderado de la imagen posterior en orden de salida es mayor que el valor ponderado de la primera de las dos.

"Superposición" indica que algunos de los valores de las muestras de cada imagen objetivo (antes de la codificación) se generaron copiando valores de muestras cúbicadas de una imagen en la escena actual y que el resto de los valores de muestras de cada imagen objetivo (antes de la codificación) se generaron copiando valores de muestras cúbicadas de una imagen en la siguiente escena. Cuando dos imágenes están etiquetadas como pertenecientes a la misma transición de escena y su `scene_transition_type` es "superposición", el número de muestras copiadas de la siguiente escena a la imagen posterior en el orden de salida es mayor que el número de muestras copiadas de la siguiente escena a la imagen anterior.

**second\_scene\_id** identifica la siguiente escena en la transición gradual de escena en la que intervienen las imágenes objetivo. El valor de `second_scene_id` será distinto del valor de `scene_id`. El valor de `second_scene_id` será distinto del valor de `scene_id` de la imagen anterior en orden de salida. Cuando la siguiente imagen en orden de salida está marcada con un valor de `scene_transition_type` menor que 4, y el valor de `second_scene_id` es el mismo que el de `scene_id` de la siguiente imagen en orden de salida, significa que el codificador considera que una de las escenas fuente de las imágenes objetivo y la escena fuente de la siguiente imagen (en orden de salida) son la misma escena. Cuando el valor de `second_scene_id` es distinto del valor de `scene_id` o `second_scene_id` (si está incluido) de la siguiente imagen en orden de salida, significa que el codificador considera que las imágenes objetivo y la siguiente imagen (en el orden de salida) pertenecen a diferentes escenas fuente.

Cuando el valor de `scene_id` de una imagen es igual al valor de `scene_id` de la siguiente imagen en orden de salida y el valor de `scene_transition_type` en ambas imágenes es menor que 4, significa que el codificador considera que las dos imágenes pertenecen a la misma escena fuente. Cuando los valores de `scene_id`, `scene_transition_type` y `second_scene_id` (si está incluida) de una imagen son iguales a los valores de `scene_id`, `scene_transition_type` y `second_scene_id` (respectivamente) de la siguiente imagen en orden de salida y el valor de `scene_transition_type` es mayor que 0, significa que el codificador considera que las dos imágenes pertenecen a la misma transición gradual de escena fuente.

El valor de `second_scene_id` estará entre 0 y  $2^{32}-1$ , inclusive. La utilización de los valores de `second_scene_id` entre 0 y 255, inclusive, y entre 512 y  $2^{31}-1$ , inclusive, depende de la aplicación. Los valores de `second_scene_id` entre 256 y 511, inclusive, y entre  $2^{31}$  y  $2^{32}-1$ , inclusive, están reservados para que el UIT-T | ISO/CEI los utilice en el futuro. Los decodificadores que detecten un

valor de `second_scene_id` entre 256 y 511, inclusive, o entre  $2^{31}$  a  $2^{32}-1$ , inclusive, no lo tendrán en cuenta (se extrae del tren de bits y se descarta).

### D.2.11 Semántica del mensaje SEI de información de subsecuencia

El mensaje SEI de información de subsecuencia se utiliza para indicar la posición de una imagen en la jerarquía de dependencia de datos que consiste en capas subsecuencia y subsecuencias.

Una capa subsecuencia incluye un subconjunto de las imágenes codificadas en una secuencia. Las capas subsecuencia se enumeran con enteros positivos. Una capa con un número más grande es una capa superior a una con un número más pequeño. Las capas se ordenan jerárquicamente basándose en su relación de dependencia de manera que ninguna imagen de una capa se predecirá utilizando una imagen de una capa superior.

NOTA – Es decir, para predecir una imagen de la capa 0 no se debe utilizar una imagen de la capa 1 o de una capa superior; en cambio, las imágenes de la capa 1 pueden predecirse a partir de imágenes de la capa 0, las imágenes en la capa 2 pueden predecirse mediante imágenes de las capas 0 y 1, etc.

NOTA – Se prevé que la calidad subjetiva aumentará con el número de capas decodificadas.

Una subsecuencia es un conjunto de imágenes codificadas dentro de una capa subsecuencia. Una imagen residirá en una sola capa subsecuencia y en una sola subsecuencia. Ninguna imagen de una subsecuencia se predecirá a partir de una imagen de otra subsecuencia de la misma capa o de una capa subsecuencia superior. Una subsecuencia de la capa 0 puede decodificarse sin utilizar imágenes que no pertenezcan a esa subsecuencia.

El mensaje SEI de información de subsecuencia se aplica a la unidad de acceso actual. En este documento la imagen codificada primaria en la unidad de acceso se denomina imagen actual.

No habrá mensaje SEI de información de subsecuencia a menos que `gaps_in_frame_num_value_allowed_flag` en el conjunto de parámetros secuencia referenciado por la imagen asociada al mensaje SEI de subsecuencia sea igual a 1.

**sub\_seq\_layer\_num** indica el número de capa subsecuencia de la imagen actual. Cuando `sub_seq_layer_num` es mayor que 0, no se utilizarán operaciones de control de gestión de memoria en ninguna cabecera de sector de la imagen actual. Cuando la imagen actual esté en una subsecuencia cuya primera imagen en orden de decodificación sea una imagen IDR, el valor de `sub_seq_layer_num` será igual a 0. En el caso de un campo de referencia desapareado, el valor de `sub_seq_layer_num` será igual a 0. El valor de `sub_seq_layer_num` estará entre 0 y 255, inclusive.

**sub\_seq\_id** indica la subsecuencia dentro de una capa. Cuando la imagen actual esté en una subsecuencia cuya primera imagen en orden de decodificación sea una imagen IDR, el valor de `sub_seq_id` será el mismo que el valor de `idr_pic_id` de la imagen IDR. El valor de `sub_seq_id` estará entre 0 y 65535, inclusive.

**first\_ref\_pic\_flag** igual a 1 indica que la imagen actual es la primera imagen de referencia de la subsecuencia en orden de decodificación. Si la imagen actual no es la primera imagen de la subsecuencia en orden de decodificación, `first_ref_pic_flag` será igual a 0.

**leading\_non\_ref\_pic\_flag** igual a 1 indica que la imagen actual es una imagen que no es de referencia y que antecede a cualquier imagen de referencia en orden de decodificación de la subsecuencia o que la subsecuencia no contiene imágenes de referencia. Cuando la imagen actual es una imagen de referencia o es una imagen que no es referencia siguiente a al menos una imagen de referencia en orden de decodificación de la subsecuencia, `leading_non_ref_pic_flag` será igual a 0.

**last\_pic\_flag** igual a 1 indica que la imagen actual es la última imagen de la subsecuencia (en orden de decodificación), incluidas todas las imágenes de referencia y las que no son referencia de la subsecuencia. Cuando la imagen actual no es la última imagen de la subsecuencia (en orden de decodificación), `last_pic_flag` será igual a 0.

La imagen actual se asigna a una subsecuencia del modo siguiente:

- Si se cumple alguna de las siguientes condiciones, la imagen actual es la primera imagen de una subsecuencia en orden de decodificación:
  - ninguna imagen anterior en orden de decodificación está etiquetada con los mismos valores de `sub_seq_id` y `sub_seq_layer_num` que la imagen actual;
  - el valor de `leading_non_ref_pic_flag` es igual a 1 y el valor de `leading_non_ref_pic_flag` es igual a 0 en la imagen anterior en orden de decodificación que tiene los mismos valores de `sub_seq_id` y `sub_seq_layer_num` que la imagen actual;
  - el valor de `first_ref_pic_flag` es igual a 1 y el valor de `leading_non_ref_pic_flag` es igual a 0 en la imagen anterior en orden de decodificación que tiene los mismos valores de `sub_seq_id` y `sub_seq_layer_num` que la imagen actual;
  - el valor de `last_pic_flag` es igual a 1 en la imagen anterior en orden de decodificación que tiene los mismos valores de `sub_seq_id` y `sub_seq_layer_num` que la imagen actual.
- De lo contrario, la imagen actual pertenece a la misma subsecuencia que la imagen anterior en orden de decodificación cuyos valores de `sub_seq_id` y `sub_seq_layer_num` son iguales a los de la imagen actual.

`sub_seq_frame_num_flag` igual a 0 indica que no hay `sub_seq_frame_num`.  
`sub_seq_frame_num_flag` igual a 1 indica que sí hay `sub_seq_frame_num`.

`sub_seq_frame_num` será igual a 0 en la primera imagen de referencia de la subsecuencia y en cualquier imagen que no sea referencia que anteceda a la primera imagen de referencia de la subsecuencia en orden de decodificación. `sub_seq_frame_num` cumple además las siguientes restricciones:

- Si la imagen actual no es el segundo campo de un par de campos complementarios, `sub_seq_frame_num` se incrementará en 1, en una operación módulo con `MaxFrameNum`, con relación a la imagen de referencia anterior, en orden de decodificación, que pertenece a la subsecuencia.
- De lo contrario (la imagen actual es el segundo campo de un par de campos complementarios) el valor de `sub_seq_frame_num` será el mismo que el valor `sub_seq_frame_num` del primer campo del par de campos complementarios.

El valor de `sub_seq_frame_num` estará entre 0 y `MaxFrameNum` – 1, inclusive.

Cuando la imagen actual es una imagen IDR, se iniciará una nueva subsecuencia en la capa subsecuencia 0. Por consiguiente, `sub_seq_layer_num` será 0, `sub_seq_id` será diferente de la subsecuencia anterior en la capa subsecuencia 0, `first_ref_pic_flag` será 1, y `leading_non_ref_pic_flag` será igual a 0.

Cuando el mensaje SEI de información de subsecuencia esté incluido en ambos campos codificados de un par de campos complementarios, los valores de `sub_seq_layer_num`, `sub_seq_id`, `leading_non_ref_pic_flag` y `sub_seq_frame_num`, si los hubiere, serán idénticos para ambas imágenes.

Cuando el mensaje SEI de información de subsecuencia esté incluido sólo en un campo codificado de un par de campos complementarios, los valores de `sub_seq_layer_num`, `sub_seq_id`, `leading_non_ref_pic_flag` y `sub_seq_frame_num`, si los hubiere, se podrán aplicar también al otro campo codificado del par de campo complementarios.

## D.2.12 Semántica del mensaje SEI de características de capa subsecuencia

El mensaje SEI de características de capa subsecuencia indica las características de las capas subsecuencia.

**num\_sub\_seq\_layers\_minus1** más 1 indica el número de capas subsecuencia en la secuencia. **num\_sub\_seq\_layers\_minus1** estará entre 0 y 255, inclusive.

Cada capa subsecuencia se caracteriza por un par de **average\_bit\_rate** y **average\_frame\_rate**. El primer par de **average\_bit\_rate** y **average\_frame\_rate** especifica las características de la capa subsecuencia 0. El segundo par, si lo hubiere, especifica las características de las capas subsecuencia 0 y 1 conjuntamente. Cada par en orden de decodificación especifica las características de una gama de capas subsecuencia, desde la capa número 0 hasta la capa con el número especificado por el contador de ciclos de capa. Los valores son válidos desde el punto en que se decodifican hasta que se decodifica una actualización de los mismos.

**accurate\_statistics\_flag** igual a 1 indica que los valores de **average\_bit\_rate** y **average\_frame\_rate** se redondearon a partir de valores correctos estadísticamente. **accurate\_statistics\_flag** igual a 0 indica que **average\_bit\_rate** y **average\_frame\_rate** son estimaciones y que pueden diferir ligeramente de los valores correctos.

**accurate\_statistics\_flag** igual a 0 indica que la precisión de la aproximación utilizada para calcular los valores de **average\_bit\_rate** y **average\_frame\_rate** fue elegida en el proceso de codificación y no se especifica en esta Recomendación | Norma Internacional.

**average\_bit\_rate** indica la velocidad binaria promedio en unidades de 1000 bits por segundo. Todas las unidades NAL en la gama de las capas subsecuencia especificadas anteriormente se tendrán en consideración en el cálculo. La velocidad binaria promedio se calcula de acuerdo con el instante de extracción de la unidad de acceso especificado en el anexo C a esta Recomendación | Norma Internacional. En lo sucesivo, **bTotal** es el número de bits de todas las unidades NAL siguientes a un mensaje SEI de características de capa subsecuencia (contando los bits de las unidades NAL de la unidad de acceso actual) y que preceden a la siguiente unidad de acceso (en orden de decodificación) incluido un mensaje SEI de características de capa subsecuencia (si lo hubiere) o del fin de tren (de lo contrario).  $t_1$  es el instante de extracción (en segundos) de la unidad de acceso actual, y  $t_2$  es el instante de extracción (en segundos) de la última unidad de acceso (en orden de decodificación) antes del subsiguiente mensaje SEI de características de capa subsecuencia (si lo hubiere) o del fin de tren (de lo contrario).

Cuando **accurate\_statistics\_flag** es igual a 1, se cumplirán las siguientes condiciones:

- Si  $t_1$  es distinto de  $t_2$ , la siguiente condición será verdadera

$$\text{average\_bit\_rate} == \text{Round}( \text{bTotal} \div ( ( t_2 - t_1 ) * 1000 ) ) \quad (\text{D-6})$$

- De lo contrario ( $t_1$  es igual a  $t_2$ ), la siguiente condición será verdadera

$$\text{average\_bit\_rate} == 0 \quad (\text{D-7})$$

**average\_frame\_rate** indica la velocidad de cuadro promedio en unidades de cuadros/(256 segundos). En el cálculo se tienen en cuenta todas las unidades NAL en la gama de las capas subsecuencia especificadas anteriormente. En lo sucesivo, **fTotal** es el número de cuadros, pares de campos complementarios y campos desapareados entre la imagen actual (inclusive) y el siguiente mensaje SEI de características de capa subsecuencia (si lo hubiere) o del fin de tren (de lo contrario).  $t_1$  es el instante de extracción (en segundos) de la unidad de acceso actual, y  $t_2$  es el instante de extracción (en segundos) de la última unidad de acceso (en orden de decodificación) antes del siguiente mensaje SEI de características de capa subsecuencia (si lo hubiere) o del fin de tren (de lo contrario).

Cuando `accurate_statistics_flag` es igual a 1, se cumplirán las siguientes condiciones:

- Si  $t_1$  es distinto de  $t_2$ , será verdadera la siguiente condición:

$$\text{average\_frame\_rate} == \text{Round}(f_{\text{Total}} * 256 \div (t_2 - t_1)) \quad (\text{D-8})$$

- De lo contrario ( $t_1$  es igual a  $t_2$ ), será verdadera la siguiente condición:

$$\text{average\_frame\_rate} == 0 \quad (\text{D-9})$$

### D.2.13 Semántica del mensaje SEI de características de subsecuencia

El mensaje SEI de características de subsecuencia indica las características de una subsecuencia. Además, indica las relaciones de dependencia de predicción inter entre las subsecuencias. Este mensaje estará contenido en la primera unidad de acceso en orden de decodificación de la subsecuencia a la que se aplica el mensaje SEI de características de subsecuencia. En este documento esta subsecuencia se denomina subsecuencia objetivo.

**sub\_seq\_layer\_num** indica el número de capa subsecuencia de la subsecuencia objetivo. El valor de `sub_seq_layer_num` estará entre 0 y 255, inclusive.

**sub\_seq\_id** indica la subsecuencia objetivo. El valor de `sub_seq_id` estará entre 0 y 65535, inclusive.

**duration\_flag** igual a 0 indica que no se ha especificado la duración de la subsecuencia objetivo.

**sub\_seq\_duration** indica la duración de la subsecuencia objetivo en tics de un reloj de 90 kHz.

**average\_rate\_flag** igual a 0 indica que no se han especificado la velocidad binaria promedio y la velocidad de cuadro promedio de la subsecuencia objetivo.

**accurate\_statistics\_flag** indica el grado de fiabilidad de los valores de `average_bit_rate` y `average_frame_rate`. `accurate_statistics_flag` igual a 1, indica que `average_bit_rate` y `average_frame_rate` se han redondeado de valores correctos estadísticamente. `accurate_statistics_flag` igual a 0 indica que `average_bit_rate` y `average_frame_rate` son valores estimados y que pueden diferir de los valores correctos estadísticamente.

**average\_bit\_rate** indica la velocidad binaria promedio en (1000 bits)/segundo de la subsecuencia objetivo. En el cálculo se tienen en cuenta todas las unidades NAL de la subsecuencia objetivo. La velocidad binaria promedio se calcula de acuerdo con el instante de extracción de la unidad de acceso conforme a la subcláusula C.1.2. En lo sucesivo,  $nB$  es el número de bits en todas las unidades NAL en la subsecuencia.  $t_1$  es el instante de extracción (en segundos) de la primera unidad de acceso de la subsecuencia (en orden de decodificación), y  $t_2$  es el instante de extracción (en segundos) de la última unidad de acceso de la subsecuencia (en orden de decodificación).

Si `accurate_statistics_flag` es igual a 1, se cumplirán las siguientes condiciones:

- Si  $t_1$  es distinto de  $t_2$ , la siguiente condición será verdadera:

$$\text{average\_bit\_rate} == \text{Round}(nB \div ((t_2 - t_1) * 1000)) \quad (\text{D-10})$$

- De lo contrario ( $t_1$  es igual a  $t_2$ ), la siguiente condición será verdadera:

$$\text{average\_bit\_rate} == 0 \quad (\text{D-11})$$

**average\_frame\_rate** indica la velocidad de cuadro promedio en unidades de cuadros/(256 segundos) de la subsecuencia objetivo. En el cálculo se tienen en cuenta todas las unidades NAL de la subsecuencia objetivo. La velocidad de cuadro promedio se deduce de acuerdo con el instante de extracción de la unidad de acceso conforme a la subcláusula C.1.2. En lo sucesivo,  $fC$  es el número de cuadros, pares de campos complementarios y campos desapareados en la subsecuencia.  $t_1$  es el instante de extracción (en segundos) de la primera unidad de acceso de la subsecuencia (en orden de decodificación), y  $t_2$  es el instante de extracción (en segundos) de la última unidad de acceso de la subsecuencia (en orden de decodificación).

Si `accurate_statistics_flag` es igual a 1, se cumplirán las siguientes condiciones:

- Si  $t_1$  es distinto de  $t_2$ , la siguiente condición será verdadera  

$$\text{average\_frame\_rate} == \text{Round}(fC * 256 \div (t_2 - t_1)) \quad (\text{D-12})$$

- De lo contrario ( $t_1$  es igual a  $t_2$ ), la siguiente condición será verdadera  

$$\text{average\_frame\_rate} == 0 \quad (\text{D-13})$$

**num\_referenced\_subseqs** indica el número de subsecuencias que contienen imágenes que se utilizan como imágenes de referencia para la predicción inter en las imágenes de la subsecuencia objetivo. El valor de num\_referenced\_subseqs estará entre 0 y 255, inclusive.

**ref\_sub\_seq\_layer\_num**, **ref\_sub\_seq\_id** y **ref\_sub\_seq\_direction** identifican la subsecuencia que contiene imágenes que se utilizan como imágenes de referencia para la predicción inter en las imágenes de la subsecuencia objetivo. En función de ref\_sub\_seq\_direction, se aplica lo siguiente:

- Si ref\_sub\_seq\_direction es igual a 0, el conjunto de subsecuencias posibles consta de las subsecuencias cuyo sub\_seq\_id es igual a ref\_sub\_seq\_id, que se encuentra en la capa subsecuencia que tiene sub\_seq\_layer\_num igual a ref\_sub\_seq\_layer\_num, y cuya primera imagen en orden de decodificación precede a la primera imagen de la subsecuencia objetivo en orden de decodificación.
- De lo contrario (ref\_sub\_seq\_direction es igual a 1), el conjunto de subsecuencias candidatas consta de las subsecuencias cuyo sub\_seq\_id es igual a ref\_sub\_seq\_id, que se encuentra en la capa subsecuencia que tiene sub\_seq\_layer\_num igual a ref\_sub\_seq\_layer\_num, y cuya primera imagen en orden de decodificación sigue a la primera imagen de la subsecuencia objetivo en orden de decodificación.

La subsecuencia utilizada como una referencia para la subsecuencia objetivo es la subsecuencia del conjunto de subsecuencias posibles cuya primera imagen es la más próxima a la primera imagen de la subsecuencia objetivo en orden de decodificación.

#### D.2.14 Semántica del mensaje SEI de congelación de todo el cuadro

El mensaje SEI de congelación de cuadro completo indica que el contenido de todo el cuadro de vídeo visualizado previamente en orden de salida debe mantenerse sin modificación, y que no se debe actualizar la visualización utilizando el contenido de la imagen decodificada actual.

**full\_frame\_freeze\_repetition\_period** indica si habrá otro mensaje SEI de congelación de todo el cuadro en el tren de bits e indica el intervalo de número de orden de imágenes en el que habrá otro mensaje SEI de congelación de todo el cuadro o un mensaje SEI de descongelación de todo el cuadro. El valor de full\_frame\_freeze\_repetition\_period estará entre 0 y 16 384, inclusive.

full\_frame\_freeze\_repetition\_period igual a 0 indica que el mensaje SEI de congelación de todo el cuadro se aplica únicamente a la imagen decodificada actual.

full\_frame\_freeze\_repetition\_period igual a 1 indica que el mensaje SEI de congelación de todo el cuadro sigue siendo aplicable en orden de salida hasta que se cumpla alguna de las siguientes condiciones:

- Comienza una nueva secuencia de vídeo codificado.
- Sale una imagen en una unidad de acceso que contiene un mensaje SEI de descongelación de todo el cuadro con PicOrderCnt() mayor que PicOrderCnt( CurrPic ).

full\_frame\_freeze\_repetition\_period mayor que 1 indica que el mensaje SEI de congelación de todo el cuadro continúa hasta que cualquiera de las siguientes condiciones sea verdadera:

- Comienza una nueva secuencia de vídeo codificado.
- Sale una imagen en una unidad de acceso que contiene un mensaje SEI de congelación de todo el cuadro o un mensaje SEI de descongelación de todo el cuadro con

PicOrderCnt( ) mayor que PicOrderCnt( CurrPic ) + full\_frame\_freeze\_repetition\_period.

full\_frame\_freeze\_repetition\_period mayor que 1 indica que habrá otro mensaje SEI de congelación de todo el cuadro o un mensaje SEI de descongelación de todo el cuadro de una imagen en una unidad de acceso que sale y que tiene PicOrderCnt( ) menor o igual que PicOrderCnt( CurrPic ) + full\_frame\_freeze\_repetition\_period, a menos que comience una nueva secuencia de vídeo codificado sin que salga esa imagen.

#### D.2.15 Semántica del mensaje SEI de descongelación de todo el cuadro

El mensaje SEI de descongelación de todo el cuadro indica que debe proseguir la visualización de cuadros de vídeo, comenzando con el contenido de la imagen decodificada actual y prosiguiendo con las imágenes subsiguientes en orden de salida. El mensaje SEI de descongelación de todo el cuadro cancela el efecto de cualquier mensaje SEI de congelación de todo el cuadro enviado con imágenes anteriores a la imagen actual en orden de salida.

#### D.2.16 Semántica del mensaje SEI de captura de todo el cuadro

El mensaje SEI de captura de todo el cuadro indica que el cuadro actual se ha etiquetado como una imagen fija capturada del contenido de vídeo, que cada aplicación utilizará.

**snapshot\_id** indica un número de identificación de la imagen capturada. El valor de snapshot\_id estará entre 0 y  $2^{32}-1$ , inclusive.

La utilización de los valores de snapshot\_id entre 0 y 255, inclusive, y entre 512 y  $2^{31}-1$ , inclusive, depende de la aplicación. Los valores de snapshot\_id entre 256 y 511, inclusive, y entre  $2^{31}$  y  $2^{32}-1$ , inclusive, están reservados para que el UIT-T | ISO/CEI los utilice en el futuro. Los decodificadores que detecten un valor de snapshot\_id entre 256 y 511, inclusive, o entre  $2^{31}$  y  $2^{32}-1$ , inclusive, no lo tendrán en cuenta (se extrae del tren de bits y se descarta).

#### D.2.17 Semántica del mensaje SEI de comienzo de segmento de refinamiento progresivo

El mensaje SEI de comienzo de segmento de refinamiento progresivo indica el inicio de un conjunto de imágenes codificadas consecutivas que está etiquetado como la imagen actual seguida por una secuencia de una o varias imágenes de refinamiento de la calidad de la imagen actual, en lugar de una representación de una escena con movimiento continuo.

El conjunto etiquetado de imágenes codificadas consecutivas continuará hasta que se cumpla alguna de las siguientes condiciones. Cuando se cumpla alguna de las siguientes condiciones, el siguiente sector que se decodificará no será uno del conjunto etiquetado de imágenes codificadas consecutivas:

1. El siguiente sector que va a decodificarse pertenece a una imagen IDR.
2. num\_refinement\_steps\_minus1 es mayor que 0 y frame\_num del siguiente sector que va a decodificarse es  $(currFrameNum + num\_refinement\_steps\_minus1 + 1) \% MaxFrameNum$ , siendo currFrameNum el valor de frame\_num de la imagen en la unidad de acceso que contiene el mensaje SEI.
3. num\_refinement\_steps\_minus1 es 0 y se decodifica un mensaje SEI de fin de segmento de refinamiento progresivo con el mismo progressive\_refinement\_id de este mensaje SEI.

El orden de decodificación de la imagen dentro del conjunto rotulado de imágenes consecutivas debería ser el mismo que su orden de salida. **progressive\_refinement\_id** indica un número de identificación para la operación de refinamiento progresivo. El valor de progressive\_refinement\_id estará entre 0 y  $2^{32}-1$ , inclusive.

La utilización de los valores de progressive\_refinement\_id entre 0 y 255, inclusive, y entre 512 y  $2^{31}-1$ , inclusive, depende de la aplicación. Los valores de progressive\_refinement\_id entre 256 y 511, inclusive, y entre  $2^{31}$  y  $2^{32}-1$ , inclusive, están reservados para que el UIT-T | ISO/CEI



los utilice en el futuro. Los decodificadores que detecten un valor de `progressive_refinement_id` entre 256 y 511, inclusive, o entre  $2^{31}$  y  $2^{32} - 1$ , inclusive, no lo tendrán en cuenta (se extrae del tren de bits y se descarta).

**num\_refinement\_steps\_minus1** indica el número de cuadros de referencia en el conjunto rotulado de imágenes codificadas consecutivas de la siguiente manera:

- Si `num_refinement_steps_minus1` es igual a 0, el número de cuadros de referencia en el conjunto rotulado de imágenes codificadas consecutivas es desconocido.
- De lo contrario, el número de cuadros de referencia en el conjunto rotulado de imágenes codificadas consecutivas es igual a `num_refinement_steps_minus1 + 1`.

El valor de `num_refinement_steps_minus1` estará entre 0 y `MaxFrameNum - 1`, inclusive.

#### **D.2.18 Semántica del mensaje SEI de fin del segmento de refinamiento progresivo**

El mensaje SEI de fin del segmento de refinamiento progresivo indica el final de un conjunto de imágenes codificadas consecutivas que se ha etiquetado, mediante un mensaje SEI de comienzo del segmento de refinamiento progresivo, como una imagen inicial seguida por una secuencia de una o varias imágenes de refinamiento de la calidad de la imagen inicial, y que termina con la imagen actual.

**progressive\_refinement\_id** indica un número de identificación para la operación de refinamiento progresivo. El valor de `progressive_refinement_id` estará entre 0 y  $2^{32} - 1$ , inclusive.

El mensaje SEI de fin del segmento de refinamiento progresivo especifica el final de cualquier segmento de refinamiento progresivo que haya comenzado antes mediante un mensaje SEI de comienzo del segmento de refinamiento progresivo con el mismo valor de `progressive_refinement_id`.

La utilización de los valores de `progressive_refinement_id` entre 0 y 255, inclusive, y entre 512 y  $2^{31} - 1$ , inclusive, depende de la aplicación. Los valores de `progressive_refinement_id` entre 256 y 511, inclusive, y entre  $2^{31}$  y  $2^{32} - 1$ , inclusive, están reservados para que el UIT-T | ISO/CEI los utilice en el futuro. Los decodificadores que detecten un valor de `progressive_refinement_id` entre 256 y 511, inclusive, o entre  $2^{31}$  y  $2^{32} - 1$ , inclusive, no lo tendrán en cuenta (se extrae del tren de bits y se descarta).

#### **D.2.19 Semántica del mensaje SEI de fin del grupo de sectores con limitación de movimiento**

Este mensaje SEI indica que la predicción inter en los contornos de grupo de sectores tiene las limitaciones que se describen más adelante. El mensaje sólo aparecerá, si aparece, cuando esté asociado a una unidad de acceso IDR, conforme a la subcláusula 7.4.1.2.3.

El conjunto de imágenes objetivo de este mensaje comprende todas las imágenes codificadas primarias consecutivas en orden de decodificación comenzando con la correspondiente imagen IDR codificada primaria (inclusive) y finalizando con la siguiente imagen IDR codificada primaria (exclusive) o con la última imagen codificada primaria en el tren de bits (inclusive) en orden de decodificación cuando no haya más imágenes IDR codificadas primarias. El conjunto del grupo de sectores agrupa uno o varios grupos de sectores, identificados por el elemento sintáctico `slice_group_id[ i ]`.

Este mensaje SEI indica que, para cada imagen del conjunto de imágenes objetivo, el proceso de predicción inter tiene las siguientes limitaciones: para la predicción inter de muestras dentro del conjunto de grupos de sectores no se utiliza ningún valor de muestra ajeno al conjunto del grupo de sectores, ni ningún valor de muestra en una posición de muestra fraccionaria que se calcule utilizando uno o más valores de muestra ajenos al conjunto de grupos de sectores.

**num\_slice\_groups\_in\_set\_minus1 + 1** indica el número de grupos de sectores en el conjunto de grupos de sectores. La gama de valores posibles de `num_slice_groups_in_set_minus1` es de 0 a

num\_slice\_groups\_minus1, inclusive. La gama de valores posibles de num\_slice\_groups\_minus1 se especifica en el anexo A.

**slice\_group\_id[ i ]** identifica el grupo o grupos de sectores incluidos en el conjunto de grupos de sectores. La gama de valores posibles es de 0 a num\_slice\_groups\_in\_set\_minus1, inclusive. El tamaño del elemento sintáctico slice\_group\_id[ i ] syntax es  $\text{Ceil}(\text{Log}_2(\text{num\_slice\_groups\_minus1} + 1))$  bits.

**exact\_sample\_value\_match\_flag** igual a 0 indica que, dentro del conjunto de imágenes objetivo, cuando no se decodifican los macrobloques que no pertenecen al conjunto de grupos de sectores, el valor de cada muestra en el conjunto de grupos de sectores no tiene que ser exactamente el mismo que el valor de la misma muestra cuando se decodifican todos los macrobloques. exact\_sample\_value\_match\_flag igual a 1 indica que, dentro del conjunto de imágenes objetivo, cuando no se decodifican los macrobloques que no pertenecen al conjunto de grupos de sectores, el valor de cada muestra en el conjunto de grupos de sectores tiene que ser exactamente el mismo que el valor de la misma muestra cuando se codifican todos los macrobloques en el conjunto de imágenes objetivo.

NOTA – Cuando disable\_deblocking\_filter\_idc es igual a 2 en todos los sectores del conjunto de imágenes objetivo, exact\_sample\_value\_match\_flag debería ser 1.

**pan\_scan\_rect\_flag** igual a 0 indica que pan\_scan\_rect\_id no está incluido. pan\_scan\_rect\_flag igual a 1 indica que pan\_scan\_rect\_id está incluido.

**pan\_scan\_rect\_id** indica que el conjunto de grupos de sectores especificado abarca al menos el rectángulo de selección identificado por pan\_scan\_rect\_id en el conjunto de imágenes objetivo.

NOTA – Es posible que haya múltiples mensajes SEI motion\_constrained\_slice\_group\_set asociados a la misma imagen IDR. Por consiguiente, puede haber más de un conjunto de grupos de sectores activos dentro de un conjunto de imágenes objetivo.

NOTA – El tamaño, forma y ubicación de los grupos de sectores en el conjunto de grupos de sectores pueden variar dentro del conjunto de imágenes objetivo.

#### **D.2.20 Semántica del mensaje SEI de reserva**

Este mensaje consta de datos reservados para que el UIT-T | ISO/CEI los utilice en el futuro para que haya compatibilidad con las versiones anteriores. Los codificadores conformes a esta Recomendación | Norma Internacional no deben enviar mensajes SEI de reserva hasta que el UIT-T | ISO/CEI haya especificado la utilización de tales mensajes. Los decodificadores conformes a esta Recomendación | Norma Internacional que detecten mensajes SEI de reserva descartarán su contenido sin afectar el proceso de decodificación, excepto conforme a las futuras Recomendaciones | Normas internacionales del UIT-T | ISO/CEI.

**reserved\_sei\_message\_payload\_byte** es un byte reservado para que el UIT-T | ISO/CEI lo utilice en el futuro.

## **Anexo E**

### **Información sobre los posibles usos de vídeo**

(Este anexo es parte integrante de esta Recomendación | Norma Internacional)

En este anexo se describe la sintaxis y la semántica de los parámetros VUI de los conjuntos de parámetros secuencia.

Los parámetros VUI no son necesarios para reconstruir las muestras luma o croma mediante el proceso de decodificación. Los decodificadores conformes no están obligados a procesar esta información para ser conformes con el orden de salida de esta Recomendación | Norma Internacional (la especificación de la conformidad figura en el anexo C). Algunos parámetros VUI son necesarios para verificar la conformidad del tren de bits y la conformidad del decodificador con la temporización de salida.

En este anexo, también se cumple con la especificación de la presencia de los parámetros VUI cuando esos parámetros (o alguno de sus subconjuntos) llegan a los decodificadores (o al HRD) por otros medios que no se especifican en esta Recomendación | Norma Internacional. Cuando los parámetros VUI están incluidos en el tren de bits, deben seguir la sintaxis y la semántica descritas en las subcláusulas 7.3.2.1 y 7.4.2.1 y en este anexo. Cuando el contenido de los parámetros VUI se transporta a la aplicación por medios distintos al tren de bits, no es obligatorio utilizar la misma sintaxis especificada en este anexo para representar el contenido de los parámetros VUI. Al contar el número de bits, sólo se han de tener en cuenta los bits apropiados que realmente estén contenidos en el tren de bits.

## E.1 Sintaxis de la información sobre los posibles usos de vídeo (VUI)

### E.1.1 Sintaxis de los parámetros VUI

	<b>C</b>	<b>Descriptor</b>
vui_parameters() {		
<b>aspect_ratio_info_present_flag</b>	0	u(1)
if( aspect_ratio_info_present_flag ) {		
<b>aspect_ratio_idc</b>	0	u(8)
if( aspect_ratio_idc == Extended_SAR ) {		
<b>sar_width</b>	0	u(16)
<b>sar_height</b>	0	u(16)
}		
}		
<b>overscan_info_present_flag</b>	0	u(1)
if( overscan_info_present_flag )		
<b>overscan_appropriate_flag</b>	0	u(1)
<b>video_signal_type_present_flag</b>	0	u(1)
if( video_signal_type_present_flag ) {		
<b>video_format</b>	0	u(3)
<b>video_full_range_flag</b>	0	u(1)
<b>colour_description_present_flag</b>	0	u(1)
if( colour_description_present_flag ) {		
<b>colour_primaries</b>	0	u(8)
<b>transfer_characteristics</b>	0	u(8)
<b>matrix_coefficients</b>	0	u(8)
}		
}		
<b>chroma_loc_info_present_flag</b>	0	u(1)
if( chroma_loc_info_present_flag ) {		
<b>chroma_sample_loc_type_top_field</b>	0	ue(v)
<b>chroma_sample_loc_type_bottom_field</b>	0	ue(v)
}		
<b>timing_info_present_flag</b>	0	u(1)
if( timing_info_present_flag ) {		
<b>num_units_in_tick</b>	0	u(32)
<b>time_scale</b>	0	u(32)
<b>fixed_frame_rate_flag</b>	0	u(1)
}		
<b>nal_hrd_parameters_present_flag</b>	0	u(1)
if( nal_hrd_parameters_present_flag )		
hrd_parameters()		
<b>vcl_hrd_parameters_present_flag</b>	0	u(1)
if( vcl_hrd_parameters_present_flag )		
hrd_parameters()		
if( nal_hrd_parameters_present_flag    vcl_hrd_parameters_present_flag )		
<b>low_delay_hrd_flag</b>	0	u(1)
<b>pic_struct_present_flag</b>	0	u(1)
<b>bitstream_restriction_flag</b>	0	u(1)
if( bitstream_restriction_flag ) {		
<b>motion_vectors_over_pic_boundaries_flag</b>	0	u(1)
<b>max_bytes_per_pic_denom</b>	0	ue(v)
<b>max_bits_per_mb_denom</b>	0	ue(v)
<b>log2_max_mv_length_horizontal</b>	0	ue(v)
<b>log2_max_mv_length_vertical</b>	0	ue(v)
<b>num_reorder_frames</b>	0	ue(v)
<b>max_dec_frame_buffering</b>	0	ue(v)
}		
}		

## E.1.2 Sintaxis de los parámetros HRD

	C	Descriptor
hrd_parameters() {		
<b>cpb cnt minus1</b>	0	ue(v)
<b>bit rate scale</b>	0	u(4)
<b>cpb size scale</b>	0	u(4)
for( SchedSelIdx = 0; SchedSelIdx <= cpb cnt minus1; SchedSelIdx++) {		
<b>bit rate value minus1</b>   SchedSelIdx	0	ue(v)
<b>cpb size value minus1</b>   SchedSelIdx	0	ue(v)
<b>cbr flag</b>   SchedSelIdx	0	u(1)
}		
<b>initial cpb removal delay length minus1</b>	0	u(5)
<b>cpb removal delay length minus1</b>	0	u(5)
<b>dpb output delay length minus1</b>	0	u(5)
<b>time offset length</b>	0	u(5)
}		

## E.2 Semántica de la información sobre uso de vídeo (VUI)

### E.2.1 Semántica de los parámetros VUI

**aspect\_ratio\_info\_present\_flag** igual a 1 indica que se incluye **aspect\_ratio\_idc**. **aspect\_ratio\_info\_present\_flag** igual a 0 indica que no se incluye **aspect\_ratio\_idc**.

**aspect\_ratio\_idc** indica el valor de la relación de aspecto de muestras luma. En el cuadro E-1 se indica el significado del código. Cuando **aspect\_ratio\_idc** es **Extended\_SAR**, la relación de aspecto de muestras se representa mediante **sar\_width** y **sar\_height**. Cuando no se incluye el elemento sintáctico **aspect\_ratio\_idc**, se inferirá que **aspect\_ratio\_idc** es igual a 0.

**Cuadro E-1 – Significado del indicador de relación de aspecto de las muestras**

<b>aspect_ratio_idc</b>	<b>Relación de aspecto de muestras</b>	<b>Ejemplos de utilización (informativo)</b>
0	No especificada	
1	1:1 ("cuadrado")	cuadro de 1280x720 16:9 sin sobrecarrido cuadro de 1920x1080 16:9 sin sobrecarrido (recortado de 1920x1088) cuadro de 640x480 4:3 sin sobrecarrido
2	12:11	cuadro de 720x576 4:3 con sobrecarrido horizontal cuadro de 352x288 4:3 sin sobrecarrido
3	10:11	cuadro de 720x480 4:3 con sobrecarrido horizontal cuadro de 352x240 4:3 sin sobrecarrido
4	16:11	cuadro de 720x576 16:9 con sobrecarrido horizontal cuadro de 540x576 4:3 con sobrecarrido horizontal
5	40:33	cuadro de 720x480 16:9 con sobrecarrido horizontal cuadro de 540x480 4:3 con sobrecarrido horizontal
6	24:11	cuadro de 352x576 4:3 sin sobrecarrido cuadro de 540x576 16:9 con sobrecarrido horizontal
7	20:11	cuadro de 352x480 4:3 sin sobrecarrido cuadro de 480x480 16:9 con sobrecarrido horizontal
8	32:11	cuadro de 352x576 16:9 sin sobrecarrido
9	80:33	cuadro de 352x480 16:9 sin sobrecarrido
10	18:11	cuadro de 480x576 4:3 con sobrecarrido horizontal
11	15:11	cuadro de 480x480 4:3 con sobrecarrido horizontal
12	64:33	cuadro de 540x576 16:9 con sobrecarrido horizontal
13	160:99	cuadro de 540x480 16:9 con sobrecarrido horizontal
14..254	Reservadas	
255	Extended_SAR	

**sar\_width** indica el tamaño horizontal de la relación de aspecto de muestras (en unidades arbitrarias).

**sar\_height** indica el tamaño vertical de la relación de aspecto de muestras (en las mismas unidades que sar\_width).

sar\_width y sar\_height serán 0 o su cociente será un número no entero en esta Recomendación | Norma Internacional. Cuando aspect\_ratio\_idc es igual a 0 o sar\_width es igual a 0 o sar\_height es igual a 0, se considerará que la relación del aspecto de muestras no está especificada.

**overscan\_info\_present\_flag** igual a 1 indica que hay un overscan\_appropriate\_flag. Si overscan\_info\_present\_flag es igual a 0 o no está, el método de visualización preferido para la señal de vídeo no está especificado.

**overscan\_appropriate\_flag** igual a 1 indica que las imágenes decodificadas recortadas resultantes se pueden visualizar mediante sobrecarrido. overscan\_appropriate\_flag igual a 0 indica que las imágenes decodificadas recortadas resultantes contienen información visual importante en toda la región exterior al rectángulo de recorte de la imagen, de manera que las imágenes decodificadas recortadas resultantes no se deberían visualizar mediante sobrecarrido. Por el contrario, deberían

visualizarse utilizando una concordancia exacta entre la zona de visualización y el rectángulo de recorte, o empleando subbarrido.

NOTA – Por ejemplo, `overscan_appropriate_flag` igual a 1 podría utilizarse para programas televisivos de ocio, o para videoconferencia, y `overscan_appropriate_flag` igual a 0 para visualizar en un monitor el contenido de una cámara de seguridad.

**video\_signal\_type\_present\_flag** igual a 1 indica que `video_format`, `video_full_range_flag` y `colour_description_present_flag` están incluidos. `video_signal_type_present_flag` igual a 0, indica que `video_format`, `video_full_range_flag` y `colour_description_present_flag` no están incluidos.

**video\_format** indica la representación de las imágenes, según se especifica en el cuadro E-2, antes de su codificación conforme a esta Recomendación | Norma Internacional. Cuando no esté el elemento sintáctico `video_format`, el valor de `video_format` se inferirá igual a 5.

**Cuadro E-2 – Significado de video\_format**

<b>video_format</b>	<b>Significado</b>
0	Componente
1	PAL
2	NTSC
3	SECAM
4	MAC
5	Formato de vídeo no especificado
6	Reservado
7	Reservado

**video\_full\_range\_flag** indica el nivel de negro y la gama de las señales luma y croma calculadas a partir de las señales de las componentes analógicas  $E'_Y$ ,  $E'_{PB}$ , y  $E'_{PR}$ , del modo siguiente:

– Si `video_full_range_flag` es igual a 0,

$$Y = \text{Round}( 219 * E'_Y + 16 ) \quad (\text{E-1})$$

$$Cb = \text{Round}( 224 * E'_{PB} + 128 ) \quad (\text{E-2})$$

$$Cr = \text{Round}( 224 * E'_{PR} + 128 ) \quad (\text{E-3})$$

– De lo contrario (`video_full_range_flag` es igual a 1),

$$Y = \text{Round}( 255 * E'_Y ) \quad (\text{E-4})$$

$$Cb = \text{Round}( 255 * E'_{PB} + 128 ) \quad (\text{E-5})$$

$$Cr = \text{Round}( 255 * E'_{PR} + 128 ) \quad (\text{E-6})$$

Cuando no esté el elemento sintáctico `video_full_range_flag`, el valor de `video_full_range_flag` se inferirá igual a 0.

**colour\_description\_present\_flag** igual a 1 indica que `colour primaries`, `transfer characteristics` y `matrix coefficients` están incluidos. `colour_description_present_flag` igual a 0 indica que `colour primaries`, `transfer characteristics` y `matrix coefficients` no están incluidos.

**colour primaries** indica las coordenadas de cromaticidad de las fuentes de colores primarios, según se especifica en el cuadro E-3 en función de la definición CEI 1931 de  $x$  y  $y$  conforme a ISO/CEI 10527.

**Cuadro E-3 – Colores primarios**

Valor	Colores primarios		
0	Reservados		
1	Rec. UIT-R BT.709		
	primario	x	y
	verde	0,300	0,600
	azul	0,150	0,060
	rojo	0,640	0,330
	blanco D65	0,3127	0,3290
2	No especificados Las características de la imagen no se conocen o dependen de la aplicación		
3	Reservados		
4	Rec. UIT-R BT.470-2 Sistema M		
	primario	x	y
	verde	0,21	0,71
	azul	0,14	0,08
	rojo	0,67	0,33
	blanco C	0,310	0,316
5	Rec. UIT-R BT.470-2 Sistema B, G		
	primario	x	y
	verde	0,29	0,60
	azul	0,15	0,06
	rojo	0,64	0,33
	blanco D65	0,3127	0,3290
6	Society of Motion Picture and Television Engineers 170M		
	primario	x	y
	verde	0,310	0,595
	azul	0,155	0,070
	rojo	0,630	0,340
	blanco D65	0,3127	0,3290
7	Society of Motion Picture and Television Engineers 240M (1987)		
	primario	x	y
	verde	0,310	0,595
	azul	0,155	0,070
	rojo	0,630	0,340
	blanco D65	0,3127	0,3290
8	Película genérica (filtros de color que emplean el iluminante C)		
	primario	x	y
	verde	0,243	0,692 ( Wratten 58 )
	azul	0,145	0,049 ( Wratten 47 )
	rojo	0,681	0,319 ( Wratten 25 )
	blanco C	0,310	0,316
9-255	Reservados		



Cuando el elemento sintáctico colour primaries no se incluye, el valor de colour primaries se inferirá igual a 2 (la cromaticidad no está especificada o depende de la aplicación).

**transfer\_characteristics** indica la característica de transferencia optoelectrónica de la imagen fuente, según se especifica en el cuadro E-4, en función de la intensidad óptica lineal incidente  $L_c$  de valor analógico comprendido entre 0 y 1.

**Cuadro E-4 – Características de transferencia**

Valor	Característica de transferencia
0	Reservada
1	Rec. UIT-R BT.709 $V = 1,099 L_c^{0,45} - 0,099$ para $1 \geq L_c \geq 0,018$ $V = 4,500 L_c$ para $0,018 > L_c$
2	No especificada Las características de la imagen se desconocen o dependen de la aplicación.
3	Reservada
4	Rec. UIT-R BT.470-2 sistema M Se supone que el gamma del monitor es 2.2
5	Rec. UIT-R BT.470-2 sistema B, G Se supone que el gamma del monitor es 2.8
6	Society of Motion Picture and Television Engineers 170M $V = 1,099 L_c^{0,45} - 0,099$ para $1 \geq L_c \geq 0,018$ $V = 4,500 L_c$ para $0,018 > L_c$
7	Society of Motion Picture and Television Engineers 240M (1987) $V = 1,1115 L_c^{0,45} - 0,1115$ para $L_c \geq 0,0228$ $V = 4,0 L_c$ para $0,0228 > L_c$
8	Características de transferencia lineal $V = L_c$
9	Característica de transferencia logarítmica (gama 100:1) $V = 1,0 - \text{Log}_{10}(L_c) \div 2$ para $1 \geq L_c \geq 0,01$ $V = 0,0$ para $0,01 > L_c$
10	Característica de transferencia logarítmica (gama 316,22777:1) $V = 1,0 - \text{Log}_{10}(L_c) \div 2,5$ para $1 \geq L_c \geq 0,0031622777$ $V = 0,0$ para $0,0031622777 > L_c$
11..255	Reservada

Cuando no se incluye el elemento sintáctico transfer\_characteristics, su valor se inferirá igual a 2 (las características de transferencia no están especificadas o las determina la aplicación).

**matrix\_coefficients** describe los coeficientes de la matriz que se utiliza para calcular las señales luma y croma a partir de los colores primarios, verde, azul o rojo, conforme al cuadro E-5.

Se utilizan las siguientes definiciones:

$E'_R$ ,  $E'_G$ , y  $E'_B$  son magnitudes analógicas de valor comprendido entre 0 y 1.

El blanco se define como  $E'_R$  igual a 1,  $E'_G$  igual a 1, y  $E'_B$  igual a 1.

Por consiguiente:

$$E'_Y = K_R * E'_R + (1 - K_R - K_B) * E'_G + K_B * E'_B \quad (E-7)$$

$$E'_{PB} = 0,5 * (E'_B - E'_Y) \div (1 - K_B) \quad (E-8)$$

$$E'_{PR} = 0,5 * (E'_R - E'_Y) \div (1 - K_R) \quad (E-9)$$

NOTA – Por consiguiente,  $E'_Y$  es una magnitud analógica cuyo valor está comprendido entre 0 y 1;  $E'_{PB}$  y  $E'_{PR}$  son magnitudes analógicas cuyos valores están entre -0,5 y 0,5, y el blanco está dado de manera similar por  $E'_Y = 1$ ,  $E'_{PB} = 0$ ,  $E'_{PR} = 0$ .

**Cuadro E-5 – Coeficientes de la matriz**

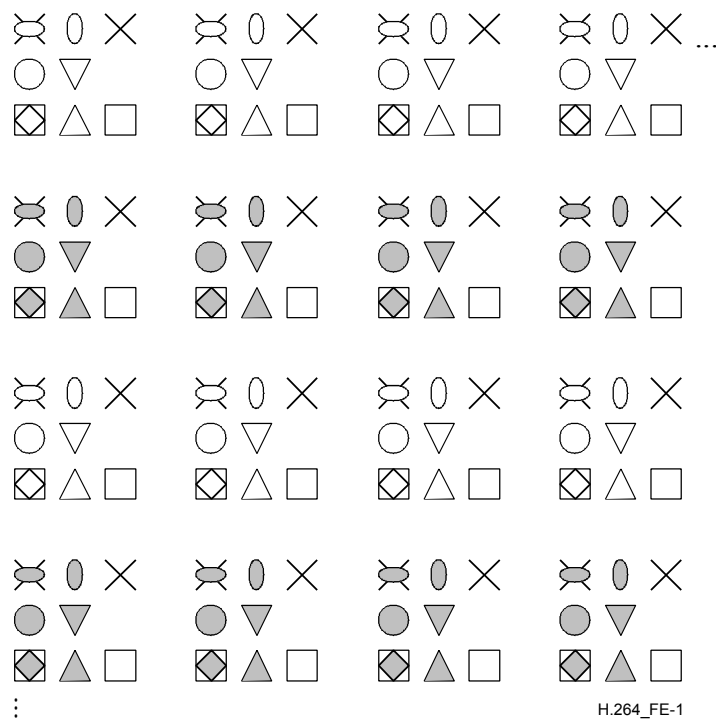
Valor	Matriz
0	Reservada
1	Rec. UIT-R BT.709 $K_R = 0,2126$ ; $K_B = 0,0722$
2	No especificada Las características de la imagen se desconocen o las determina la aplicación.
3	Reservada
4	Federal Communications Commission $K_R = 0,30$ ; $K_B = 0,11$
5	Rec. UIT-R BT.470-2 Sistema B, G: $K_R = 0,299$ ; $K_B = 0,114$
6	Society of Motion Picture and Television Engineers 170M $K_R = 0,299$ ; $K_B = 0,114$
7	Society of Motion Picture and Television Engineers 240M (1987) $K_R = 0,212$ ; $K_B = 0,087$
8-255	Reservada

Cuando no está incluido el elemento de sintaxis `matrix_coefficients`, su valor se inferirá igual a 2.

**chroma\_loc\_info\_present\_flag** igual a 1 indica que `chroma_sample_loc_type_top_field` y `chroma_sample_loc_type_bottom_field` están incluidos. `chroma_loc_info_present_flag` igual a 0 indica que `chroma_sample_loc_type_top_field` y `chroma_sample_loc_type_bottom_field` no están incluidos.

**chroma\_sample\_loc\_type\_top\_field** y **chroma\_sample\_loc\_type\_bottom\_field** indican la ubicación de las muestras de croma del campo superior y del campo inferior, como se muestra en la figura E-1. Los valores de `chroma_sample_loc_type_top_field` y `chroma_sample_loc_type_bottom_field` estarán entre 0 y 5, inclusive. Cuando `chroma_sample_loc_type_top_field` y `chroma_sample_loc_type_bottom_field` no están incluidos, sus valores se inferirán igual a 0.

NOTA – Cuando se codifica material fuente progresivo, `chroma_sample_loc_type_top_field` y `chroma_sample_loc_type_bottom_field` deben tener el mismo valor.



Interpretación de símbolos:

Indicaciones de la posición de muestras luma:

Campo superior de muestras luma    Campo inferior de muestras luma

Indicaciones de la posición de muestras croma, donde el relleno en gris indica un tipo de muestra de campo inferior y sin relleno indica un tipo de muestra de campo superior:

Tipo de muestra croma 2                      Tipo de muestra croma 3  
 Tipo de muestra croma 0                      Tipo de muestra croma 1  
 Tipo de muestra croma 4                      Tipo de muestra croma 5

**Figura E-1 – Ubicación de las muestras croma de los campos superior e inferior en función de chroma\_sample\_loc\_type\_top\_field y chroma\_sample\_loc\_type\_bottom\_field**

**timing\_info\_present\_flag** igual a 1 indica que el tren de bits contiene num\_units\_in\_tick, time\_scale y fixed\_frame\_rate\_flag. timing\_info\_present\_flag igual a 0 indica que num\_units\_in\_tick, time\_scale y fixed\_frame\_rate\_flag no están incluidos en el tren de bits.

**num\_units\_in\_tick** es el número de unidades de tiempo de un reloj de frecuencia time\_scale Hz que corresponde al incremento en una unidad (denominada tic de reloj) de un contador de tics de reloj. num\_units\_in\_tick será mayor que 0. Un tic de reloj es el intervalo mínimo de tiempo que puede representarse en los datos codificados. Por ejemplo, cuando la frecuencia de reloj de una señal de vídeo es 30 000 ÷ 1 001 Hz, time\_scale podría ser 30 000 y num\_units\_in\_tick podría ser 1 001. Véase la ecuación (C-1).

**time\_scale** es el número de unidades de tiempo que transcurren en un segundo. Por ejemplo, un sistema controlado por tiempo que mide el tiempo con un reloj de 27 MHz tiene un time\_scale de 27 000 000. time\_scale será mayor que 0.

**fixed\_frame\_rate\_flag** igual a 1 indica que el intervalo de tiempo entre los tiempos de salida del HRD de dos imágenes consecutivas en orden de salida tiene las siguientes restricciones. fixed\_frame\_rate\_flag igual a 0 indica que no se aplican restricciones al intervalo de tiempo entre los instantes de salida del HRD de dos imágenes consecutivas cualesquiera en el orden de salida.

Cuando `fixed_frame_rate_flag` es igual a 1, para todo `n`, siendo `n` la `n`ésima imagen en el orden de salida y siendo distinta de la última imagen del tren de bits en el orden de salida, el valor de  $\Delta t_{fi,dpb}(n)$  se calcula mediante:

$$\Delta t_{fi,dpb}(n) = \Delta t_{o,dpb}(n) \div \text{DeltaTfiDivisor} \quad (\text{E-10})$$

donde  $\Delta t_{o,dpb}(n)$  se especifica en la ecuación C-13 y `DeltaTfiDivisor` se especifica en el cuadro E-6 en función de los valores de `pic_struct_present_flag`, `field_pic_flag`, y `pic_struct` de la imagen `n`. El símbolo "-" en el cuadro E-6 indica que `DeltaTfiDivisor` no depende del elemento sintáctico correspondiente.

Cuando `fixed_frame_rate_flag` es igual 1, el valor calculado de  $\Delta t_{fi,dpb}(n)$  será el mismo para todo `n > 0` y será igual a `num_units_in_tick`  $\div$  `time_scale`.

**Cuadro E-6 – Divisor para el cálculo de  $\Delta t_{fi,dpb}(n)$**

<code>pic_struct_present_flag</code>	<code>field_pic_flag</code>	<code>pic_struct</code>	<code>DeltaTfiDivisor</code>
0	1	–	1
1	–	1	1
1	–	2	1
0	0	–	2
1	–	0	2
1	–	3	2
1	–	4	2
1	–	5	3
1	–	6	3
1	–	7	4
1	–	8	6

`nal_hrd_parameters_present_flag` igual a 1 indica que el tren de bits contiene los parámetros HRD NAL (que pertenecen a la conformidad del tren de bits Tipo II). `nal_hrd_parameters_present_flag` igual a 0 indica que los parámetros NAL HRD no están incluidos.

NOTA – Cuando `nal_hrd_parameters_present_flag` es igual a 0, no se puede verificar la conformidad del tren de bits si no se suministran los parámetros del HRD NAL, incluida la información del parámetro del HRD de la secuencia NAL y todos los mensajes SEI de periodo de almacenamiento intermedio y de temporización de la imagen, por otros medios que no se especifican en esta Recomendación | Norma Internacional

Cuando `nal_hrd_parameters_present_flag` es igual a 1, los parámetros HRD NAL (subcláusulas E.1.2 y E.2.2) están inmediatamente a continuación del indicador.

La variable `NalHrdBpPresentFlag` se calcula del modo siguiente:

- Si cumple algunas de las siguientes condiciones, el valor de `NalHrdBpPresentFlag` se pone a 1:
  - el tren de bits contiene `nal_hrd_parameters_present_flag` y su valor es 1;
  - la aplicación determina la necesidad para el funcionamiento del HRD NAL de que haya periodos de almacenamiento intermedio en el tren de bits en los mensajes SEI de periodo de almacenamiento intermedio, por medios que no se especifican en esta Recomendación | Norma Internacional.
- De lo contrario, el valor de `NalHrdBpPresentFlag` se pone a 0.

**vcl\_hrd\_parameters\_present\_flag** igual a 1 indica que el tren de bits contiene los parámetros del HRD VCL (que pertenecen a la conformidad con todos los trenes de bits). **vcl\_hrd\_parameters\_present\_flag** igual a 0 indica que los parámetros del HRD VCL no están incluidos.

NOTA – Cuando **vcl\_hrd\_parameters\_present\_flag** es igual a 0, no se puede verificar la conformidad del tren de bits si no se suministran los parámetros del HRD VCL y todos los mensajes SEI de periodo de almacenamiento intermedio y de temporización de imagen, por otros medios que no se especifican en esta Recomendación | Norma Internacional.

Cuando **vcl\_hrd\_parameters\_present\_flag** es igual a 1, los parámetros VCL HRD (subcláusulas E.1.2 y E.2.2) están inmediatamente a continuación del indicador.

La variable **VclHrdBpPresentFlag** se calcula del modo siguiente:

- Si se cumple alguna de las siguientes condiciones, el valor de **VclHrdBpPresentFlag** se pone a 1:
  - el tren de bits incluye a **nal\_hrd\_parameters\_present\_flag** y su valor es 1;
  - la aplicación determina la necesidad para el funcionamiento del HRD VCL de que haya periodos de almacenamiento intermedio en el tren de bits en los mensajes SEI de periodo de almacenamiento intermedio, por otros medios que no se especifican en esta Recomendación | Norma Internacional.
- De lo contrario, el valor de **VclHrdBpPresentFlag** se pone a 0.

La variable **CpbDpbDelaysPresentFlag** se calcula del modo siguiente:

- Si se cumple alguna de las siguientes condiciones, el valor de **CpbDpbDelaysPresentFlag** se pone a 1:
  - el tren de bits incluye a **nal\_hrd\_parameters\_present\_flag** y su valor es 1;
  - el tren de bits incluye a **vcl\_hrd\_parameters\_present\_flag** y su valor es 1;
  - la aplicación determina la necesidad que el tren de bits contenga los retardos de salida de la CPB y la DPB en los mensajes SEI de temporización de imagen, por otros medios que no se especifican en esta Recomendación | Norma Internacional.
- De lo contrario, el valor de **CpbDpbDelaysPresentFlag** se pone a 0.

**low\_delay\_hrd\_flag** indica el modo de funcionamiento del HRD, como se describe en el anexo C. Cuando **fixed\_frame\_rate\_flag** es igual a 1, **low\_delay\_hrd\_flag** será igual a 0.

NOTA – Cuando **low\_delay\_hrd\_flag** es igual a 1, se permiten "imágenes grandes" que rebasan los tiempos nominales de extracción de la CPB debido al número de bits utilizados por una unidad de acceso. Se espera, aunque no es obligatorio, que habrá "imágenes grandes" muy de vez en cuando.

**pic\_struct\_present\_flag** igual a 1 indica que el tren de bits contiene mensajes SEI de temporización de imágenes (subcláusula D.2.2) que incluyen el elemento sintáctico **pic\_struct**. **pic\_struct\_present\_flag** igual a 0 indica que los mensajes SEI de temporización de imagen no contienen el elemento sintáctico **pic\_struct**.

**bitstream\_restriction\_flag** igual a 1, indica que el tren de bits contiene los siguientes parámetros de restricción del tren de bits de secuencia. **bitstream\_restriction\_flag** igual a 0, especifica que los siguientes parámetros de restricción de tren de bits de secuencia no están incluidos.

**motion\_vectors\_over\_pic\_boundaries\_flag** igual a 0 indica que para la predicción Inter de muestras no se utiliza ninguna muestra exterior a los bordes de la imagen y ninguna muestra en una posición de muestra fraccionaria cuyo valor se calcule utilizando una o más muestras exteriores a los bordes de la imagen. **motion\_vectors\_over\_pic\_boundaries\_flag** igual a 1 indica que para la predicción inter pueden utilizarse una o más muestras exteriores a los bordes de la imagen. Cuando el elemento sintáctico **motion\_vectors\_over\_pic\_boundaries\_flag** no esté incluido, su valor se inferirá igual a 1.

**max\_bytes\_per\_pic\_denom** indica un número de bytes no mayor a la suma de los tamaños de las unidades NAL VCL asociadas a cualquier imagen codificada en la secuencia.

Para este fin, el número de bytes que representa una imagen en el tren de unidades NAL se define como el número total de bytes de datos de la unidad NAL VCL (es decir, el total de las variables NumBytesInNALunit de las unidades NAL VCL) de la imagen. El valor de max\_bytes\_per\_pic\_denom estará entre 0 y 16, inclusive.

En función de max\_bytes\_per\_pic\_denom se aplica lo siguiente:

- Si max\_bytes\_per\_pic\_denom es igual a 0, no hay límites.
- De lo contrario (max\_bytes\_per\_pic\_denom es distinto de 0), ninguna imagen codificada se representará en la secuencia con un número de bytes mayor que el siguiente.

$$(\text{PicSizeInMbs} * 256 * \text{ChromaFormatFactor}) \div \text{max\_bytes\_per\_pic\_denom} \quad (\text{E-11})$$

Cuando el elemento sintáctico max\_bytes\_per\_pic\_denom no esté incluido, su valor se inferirá igual a 2.

**max\_bits\_per\_mb\_denom** indica el número máximo de bits codificados de datos macroblock\_layer() de cualquier macrobloque en cualquier imagen de la secuencia. El valor de max\_bits\_per\_mb\_denom estará entre 0 y 16, inclusive.

En función de max\_bits\_per\_mb\_denom se aplica lo siguiente:

- Si max\_bits\_per\_mb\_denom es igual a 0, no se especifica ningún límite.
- De lo contrario (max\_bits\_per\_mb\_denom es distinto de 0), ninguna macroblock\_layer() codificada se representará en el tren de bits con un número de bits mayor que el siguiente.

$$(2048 * \text{ChromaFormatFactor} + 128) \div \text{max\_bits\_per\_mb\_denom} \quad (\text{E-12})$$

En función de entropy\_coding\_mode\_flag, los bits de macroblock\_layer() se cuentan como se indica a continuación:

- Si entropy\_coding\_mode\_flag es igual a 0, el número de bits de datos macroblock\_layer() está dado por el número de bits en la estructura sintáctica macroblock\_layer() de un macrobloque.
- De lo contrario (entropy\_coding\_mode\_flag es igual a 1), el número de bits de datos macroblock\_layer() de un macrobloque está dado por el número de veces que se llama a read\_bits(1), las subcláusulas 9.3.3.2.2 y 9.3.3.2.3, cuando se efectúa el análisis de la macroblock\_layer() asociada al macrobloque.

Cuando el max\_bits\_per\_mb\_denom no está incluido, se inferirá que vale 1.

**log2\_max\_mv\_length\_horizontal** y **log2\_max\_mv\_length\_vertical** indican el máximo en valor absoluto de una componente de vector de movimiento horizontal y vertical decodificado, respectivamente, en unidades de 1/4 de muestra luma, de todas las imágenes en la secuencia. Para todo n se cumple que ningún valor de una componente de vector de movimiento estará por fuera de la gama comprendida entre  $-2^n$  y  $2^n-1$ , inclusive, en unidades de 1/4 de muestra luma. El valor de log2\_max\_mv\_length\_horizontal estará entre 0 y 16, inclusive. El valor de log2\_max\_mv\_length\_vertical estará entre 0 y 16, inclusive. Cuando log2\_max\_mv\_length\_horizontal no esté incluido, los valores de log2\_max\_mv\_length\_horizontal y log2\_max\_mv\_length\_vertical se inferirán iguales a 16.

NOTA – El máximo en valor absoluto de una componente de vector de movimiento vertical u horizontal decodificado está restringido también por los límites de perfil y nivel especificados en el anexo A.

**num\_reorder\_frames** indica el máximo número de cuadros, pares de campos complementarios o campos desapareados que anteceden a cualquier cuadro, par de campos complementarios o campo desapareado en la secuencia en orden de decodificación y que lo siguen en orden de salida. El valor

de `num_reorder_frames` estará entre 0 y `max_dec_frame_buffering`, inclusive. Cuando el elemento sintáctico `num_reorder_frames` no esté incluido, se inferirá que vale `max_dec_frame_buffering`.

**max\_dec\_frame\_buffering** indica el tamaño necesario de la memoria intermedia de imágenes decodificadas (DPB) del HRD en unidades de memoria intermedia de cuadros. Las secuencias no necesitarán una memoria intermedia de imágenes decodificadas con un tamaño mayor que  $\text{Max}(1, \text{max\_dec\_frame\_buffering})$  memorias intermedias de cuadros para sacar las imágenes decodificadas en los instantes de salida especificados por `dpb_output_delay` en los mensajes SEI de temporización de imágenes. El valor de `max_dec_frame_buffering` estará entre `num_ref_frames` y `MaxDpbSize` (conforme a la subcláusula A.3.1), inclusive. Cuando el elemento sintáctico `max_dec_frame_buffering` no esté incluido, se inferirá que vale `MaxDpbSize`.

## E.2.2 Semántica de los parámetros HRD

**cpb\_cnt\_minus1** más 1 indica el número de especificaciones CPB alternativas en el tren de bits. El valor de `cpb_cnt_minus1` estará entre 0 y 31, inclusive. Cuando `low_delay_hrd_flag` sea igual a 1, `cpb_cnt_minus1` será igual a 0. Cuando `cpb_cnt_minus1` no esté incluido, se inferirá que vale 0.

**bit\_rate\_scale** (junto con `bit_rate_value_minus1[ SchedSelIdx ]`) indica la máxima velocidad binaria de entrada de la `SchedSelIdx`-ésima CPB.

**cpb\_size\_scale** (junto con `cpb_size_value_minus1[ SchedSelIdx ]`) indica el tamaño de la `SchedSelIdx`-ésima CPB.

**bit\_rate\_value\_minus1[ SchedSelIdx ]** (junto con `bit_rate_scale`) indica la máxima velocidad binaria de entrada de la `SchedSelIdx`-ésima CPB. El valor de `bit_rate_value_minus1[ SchedSelIdx ]` estará entre 0 y  $2^{32}-2$ , inclusive. Para cualquier `SchedSelIdx > 0`, `bit_rate_value_minus1[ SchedSelIdx ]` deberá ser mayor que `bit_rate_value_minus1[ SchedSelIdx - 1 ]`. La velocidad binaria en bits por segundo está dada por:

$$\text{BitRate}[ \text{SchedSelIdx} ] = ( \text{bit\_rate\_value\_minus1}[ \text{SchedSelIdx} ] + 1 ) * 2^{(6 + \text{bit\_rate\_scale})} \quad (\text{E-13})$$

Cuando el elemento sintáctico `bit_rate_value_minus1[ SchedSelIdx ]` no esté incluido, el valor de `BitRate[ SchedSelIdx ]` se inferirá igual a  $1000 * \text{MaxBR}$  para los parámetros HRD VCL.

Cuando el elemento sintáctico `bit_rate_value_minus1[ SchedSelIdx ]` no esté incluido, el valor de `BitRate[ SchedSelIdx ]` se inferirá igual a  $1200 * \text{MaxBR}$  para los parámetros HRD NAL.

**cpb\_size\_value\_minus1[ SchedSelIdx ]** se utiliza junto con `cpb_size_scale` para especificar el tamaño de la `SchedSelIdx`-ésima CPB. El valor de `cpb_size_value_minus1[ SchedSelIdx ]` estará entre 0 y  $2^{32}-2$ , inclusive. Para cualquier `SchedSelIdx` mayor que cero 0, `cpb_size_value_minus1[ SchedSelIdx ]` será menor o igual que `cpb_size_value_minus1[ SchedSelIdx - 1 ]`.

El tamaño de la CPB en bits está dado por:

$$\text{CpbSize}[ \text{SchedSelIdx} ] = ( \text{cpb\_size\_value\_minus1}[ \text{SchedSelIdx} ] + 1 ) * 2^{(4 + \text{cpb\_size\_scale})} \quad (\text{E-14})$$

Cuando el elemento sintáctico `cpb_size_value_minus1[ SchedSelIdx ]` no esté incluido, el valor de `CpbSize[ SchedSelIdx ]` se inferirá igual a  $1000 * \text{MaxCPB}$  para los parámetros HRD VCL.

Cuando el elemento sintáctico `cpb_size_value_minus1[ SchedSelIdx ]` no esté incluido, el valor de `CpbSize[ SchedSelIdx ]` se inferirá igual a  $1200 * \text{MaxCPB}$  para los parámetros HRD NAL.

En los parámetros HRD VCL, habrá al menos un valor de `SchedSelIdx` para el cual  $\text{BitRate}[ \text{SchedSelIdx} ] \leq 1000 * \text{MaxBR}$  y de  $\text{CpbSize}[ \text{SchedSelIdx} ] \leq 1000 * \text{MaxCPB}$  (conforme a la subcláusula A.3.1).

En los parámetros HRD NAL, habrá al menos un valor de `SchedSelIdx` para el cual  $\text{CpbSize}[ \text{SchedSelIdx} ] \leq 1200 * \text{MaxCPB}$  y  $\text{BitRate}[ \text{SchedSelIdx} ] \leq 1200 * \text{MaxBR}$ .

**cbr\_flag**[ SchedSelIdx ] igual a 0 indica que para decodificar este tren de bits mediante el HRD utilizando la especificación de la SchedSelIdx-ésima CPB, el planificador ficticio de trenes (HSS) funciona en modo de velocidad binaria intermitente. **cbr\_flag**[ SchedSelIdx ] igual a 1 indica que el HSS funciona en modo de velocidad binaria constante (CBR). Cuando el elemento sintáctico **cbr\_flag**[ SchedSelIdx ] no esté incluido, se inferirá que **cbr\_flag** vale 0.

**initial\_cpb\_removal\_delay\_length\_minus1** indica la longitud en bits de los elementos sintácticos **initial\_cpb\_removal\_delay**[ SchedSelIdx ] e **initial\_cpb\_removal\_delay\_offset**[ SchedSelIdx ] del mensaje SEI de periodo de almacenamiento intermedio. La longitud de **initial\_cpb\_removal\_delay**[ SchedSelIdx ] y de **initial\_cpb\_removal\_delay\_offset**[ SchedSelIdx ] es **initial\_cpb\_removal\_delay\_length\_minus1** + 1. Cuando el elemento sintáctico **initial\_cpb\_removal\_delay\_length\_minus1** esté en más de una estructura sintáctica **hrd\_parameters()** dentro de la estructura sintáctica parámetros VUI, el valor de los parámetros **initial\_cpb\_removal\_delay\_length\_minus1** será idéntico en ambas estructuras sintácticas **hrd\_parameters()**. Cuando el elemento sintáctico **initial\_cpb\_removal\_delay\_length\_minus1** no esté incluido, se inferirá que vale 23.

**cpb\_removal\_delay\_length\_minus1** indica la longitud en bits del elemento sintáctico **cpb\_removal\_delay**. La longitud del elemento sintáctico **cpb\_removal\_delay** del mensaje SEI de temporización de imágenes es **cpb\_removal\_delay\_length\_minus1** + 1. Cuando el elemento sintáctico **cpb\_removal\_delay\_length\_minus1** esté en más de una estructura sintáctica **hrd\_parameters()** dentro de la estructura sintáctica parámetros VUI, el valor de los parámetros **cpb\_removal\_delay\_length\_minus1** será idéntico en ambas estructuras sintácticas **hrd\_parameters()**. Cuando el elemento sintáctico **cpb\_removal\_delay\_length\_minus1** no esté incluido, se inferirá que vale 23.

**dpb\_output\_delay\_length\_minus1** indica la longitud en bits del elemento sintáctico **dpb\_output\_delay**. La longitud del elemento sintáctico **dpb\_output\_delay** del mensaje SEI de temporización de imágenes es **dpb\_output\_delay\_length\_minus1** + 1. Cuando el elemento sintáctico **dpb\_output\_delay\_length\_minus1** esté incluido en más de una estructura sintáctica **hrd\_parameters()** dentro de la estructura sintáctica parámetros VUI, el valor de los parámetros **dpb\_output\_delay\_length\_minus1** será idéntico en ambas estructuras sintácticas **hrd\_parameters()**. Cuando el elemento sintáctico **dpb\_output\_delay\_length\_minus1** no esté incluido, se inferirá que vale 23.

**time\_offset\_length** mayor que 0 indica la longitud en bits del elemento sintáctico **time\_offset**. **time\_offset\_length** igual a 0 indica que el elemento sintáctico **time\_offset** no está incluido. Cuando el elemento sintáctico **time\_offset\_length** esté en más de una estructura sintáctica **hrd\_parameters()** dentro de la estructura sintáctica parámetros VUI, el valor de los parámetros **time\_offset\_length** será idéntico en ambas estructuras sintácticas **hrd\_parameters()**. Cuando el elemento sintáctico **time\_offset\_length** no esté incluido, se inferirá que vale 24.





## SERIES DE RECOMENDACIONES DEL UIT-T

Serie A	Organización del trabajo del UIT-T
Serie D	Principios generales de tarificación
Serie E	Explotación general de la red, servicio telefónico, explotación del servicio y factores humanos
Serie F	Servicios de telecomunicación no telefónicos
Serie G	Sistemas y medios de transmisión, sistemas y redes digitales
<b>Serie H</b>	<b>Sistemas audiovisuales y multimedia</b>
Serie I	Red digital de servicios integrados
Serie J	Redes de cable y transmisión de programas radiofónicos y televisivos, y de otras señales multimedia
Serie K	Protección contra las interferencias
Serie L	Construcción, instalación y protección de los cables y otros elementos de planta exterior
Serie M	RGT y mantenimiento de redes: sistemas de transmisión, circuitos telefónicos, telegrafía, facsímil y circuitos arrendados internacionales
Serie N	Mantenimiento: circuitos internacionales para transmisiones radiofónicas y de televisión
Serie O	Especificaciones de los aparatos de medida
Serie P	Calidad de transmisión telefónica, instalaciones telefónicas y redes locales
Serie Q	Conmutación y señalización
Serie R	Transmisión telegráfica
Serie S	Equipos terminales para servicios de telegrafía
Serie T	Terminales para servicios de telemática
Serie U	Conmutación telegráfica
Serie V	Comunicación de datos por la red telefónica
Serie X	Redes de datos y comunicación entre sistemas abiertos
Serie Y	Infraestructura mundial de la información, aspectos del protocolo Internet y Redes de la próxima generación
Serie Z	Lenguajes y aspectos generales de soporte lógico para sistemas de telecomunicación