**ISO/IEC 23008-12:2025/CDAM 2:2025(E)**

ISO/IEC JTC1/SC 29

Secretariat: JISC

**Information technology — High efficiency coding and media delivery in heterogeneous environments — Part 12: Image File Format — Amendment 2: Low-overhead image file format**

DAM stage

**Copyright notice**

This ISO document is a working draft or committee draft and is copyright-protected by ISO. While the reproduction of working drafts or committee drafts in any form for use by participants in the ISO standards development process is permitted without prior permission from ISO, neither this document nor any extract from it may be reproduced, stored or transmitted in any form for any other purpose without prior written permission from ISO.

Requests for permission to reproduce this document for the purpose of selling it should be addressed as shown below or to ISO's member body in the country of the requester:

ISO copyright office

Case postale 56 • CH-1211 Geneva 20

Tel. + 41 22 749 01 11

Fax + 41 22 749 09 47

E-mail copyright@iso.org

Web www.iso.org

Reproduction for sales purposes may be subject to royalty payments or a licensing agreement.

Violators may be prosecuted.

**Foreword**

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1.  In particular the different approval criteria needed for the different types of documents should be noted.  This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see [www.iso.org/directives](http://www.iso.org/directives) or [www.iec.ch/members\_experts/refdocs](http://www.iec.ch/members_experts/refdocs)).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see [www.iso.org/patents](http://www.iso.org/patents)) or the IEC list of patent declarations received (see [patents.iec.ch](https://patents.iec.ch/)).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see the following URL:  [www.iso.org/iso/foreword.html](http://www.iso.org/iso/foreword.html). In the IEC, see [www.iec.ch/understanding-standards](https://www.iec.ch/understanding-standards)

This document was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, SC 29, *Coding of audio, picture, multimedia and hypermedia information*.

A list of all parts in the ISO/IEC 14496 and ISO/IEC 23008 series can be found on the ISO and IEC websites.

Any feedback or questions on this document should be directed to the user’s national standards body. A complete listing of these bodies can be found at [www.iso.org/members.html](https://www.iso.org/members.html) and [www.iec.ch/national-committees](http://www.iec.ch/national-committees).

**Information technology — High efficiency coding and media delivery in heterogeneous environments — Part 12: Image File Format — Amendment 2: Low-overhead image file format**

Add the following at the end of clause 2:

Annex O specifies the low-overhead image file format suitable for small and simple images used in specific use cases. In this format, the top-level MetaBox and MediaDataBox are replaced by a MinimizedImageBox to reduce the overhead relative to the structure-data describing the image or metadata payloads, while logically representing the contents of the MetaBox and MediaDataBox. Annex O also specifies the requirements and brands for this format, as well as the procedure for expanding it to the regular Image File Format.

Add the following new definition in subclause 3.1:

**3.1.XX tiled image item**

An image item constructed using uniform tiled subregions, allowing direct access to individual tiles.

Add the following NOTE after the paragraph "Images can be stored as items using the support for untimed data storage, called the MetaBox for historical reasons, in the ISO base media file format. A file may contain any number of image items." in clause 6.1:

NOTE For small and simple files, a pre-processed version of the MetaBox called the MinimizedImageBox, as defined in Annex O, can be used. A file containing a MinimizedImageBox has the same compliance and rendering requirements as an Image File Format file because the MinimizedImageBox is expanded into the equivalent MetaBox and MediaDataBox as described in detail in Annex O.4. Once expanded, the rest of Clause 6 applies.

Change the sentence in subclause 6.4.10 from (DAM1):

An overview image is described by a grid derived image item or a tiled pre-derived coded image item whose reconstructed image is formed from generating a lower resolution, ‘binned’ version of the reconstructed image of a base image item. The base image item is also tiled. The tiling may be implemented using a feature of a specific codec, or by using a grid derived image item. When a grid derived image item is used, the input items to the grid define the tiles. Derived image items shall not be used as inputs to the image grid, due to the need for in place byte range accessing of content. Individual tiles shall be written contiguously in memory, thereby allowing access with a single read or write action.

To:

An overview image is described by a grid derived image item or a tiled pre-derived coded image item or a tiled image item whose reconstructed image is formed from generating a lower resolution, ‘binned’ version of the reconstructed image of a base image item. The base image item is also tiled. The tiling may be implemented using a feature of a specific codec, or by using a grid derived image item, or by using a tiled image item. When a grid derived image item is used, the input items to the grid define the tiles. Derived image items shall not be used as inputs to the image grid, due to the need for in place byte range accessing of content. Individual tiles shall be written contiguously in memory, thereby allowing access with a single read or write action.

Replace the third and fourth paragraphs in clause 6.5.1 with:

The semantics of the descriptive properties specified in 6.5 are specified for the image before the transformations, if any, are applied.

NOTE It is uncertain if readers would be able to correctly interpret descriptive properties that follow the first transformative property or the first unrecognized essential property, whichever is earlier, in the sequence associating properties with an item, because those descriptive properties possibly describe the output image after the transformation(s). In previous versions of this document, readers had to allow and ignore descriptive properties following the first transformative or unrecognized property, whichever is earlier, in the sequence associating properties with an item.

Writers should arrange the descriptive properties prior to any transformative property in the sequence associating properties with an item. Writers should arrange the descriptive properties specified in 6.5 prior to any other properties in the sequence associating properties with an item.

Change the following text in clause 6.5.6.1:

The PixelInformationProperty descriptive item property indicates the number and bit  
depth of colour components in the reconstructed image of the associated image item.

To:

The PixelInformationProperty descriptive item property indicates the number and bit depth of colour and alpha/depth components, if present, in the reconstructed image of the associated image item.

Change the syntax in clause 6.5.6.2 to:

aligned(8) class PixelInformationProperty

   extends ItemFullProperty('pixi', version = 0, px\_flags){

   unsigned int(8) num\_channels;

   for (i=0; i<num\_channels; i++) {

      unsigned int(8) bits\_per\_channel;

   }

   if((px\_flags & 1) != 0) {

      for (i=0; i<num\_channels; i++) {

         unsigned int(3) channel\_idc;

         unsigned int(1) reserved = 0;

         unsigned int(2) component\_format;

         unsigned int(1) subsampling\_flag;

         unsigned int(1) channel\_label\_flag;

         if(subsampling\_flag) {

            unsigned int(4) subsampling\_type;

            unsigned int(4) subsampling\_location;

         }

         if(channel\_label\_flag) {

            utf8string channel\_label;

         }

      }

   }  
}

Change the bits\_per\_channel semantic in clause 6.5.6.3 from:

bits\_per\_channel: This field indicates the bits per channel for the pixels of the reconstructed image of the associated image item.

to:

bits\_per\_channel: This field indicates the bits per channel for the pixels of the reconstructed image of the associated image item. The value of this field shall not be 0.

Add the following text to the end of clause 6.5.6.3:

px\_flags&1: If equal to 1, indicates that the channel\_idc, component\_format, and channel\_label\_flag fields are present.

channel\_idc: This field indicates the contents of the channel. A value of 0 indicates colour/grayscale. A value of 1 indicates alpha. A value of 2 indicates depth. Values 3-7 are reserved for future use. At most one channel shall have a channel\_idc of 1.

component\_format: This field indicates the data type of the channel as defined by the component\_format values in ISO/IEC 23001-17 where component\_bit\_depth is considered to be equal to bits\_per\_channel.

subsampling\_flag: If equal to 1, indicates that the subsampling\_type and subsampling\_location fields are present.

channel\_label\_flag: This flag indicates the presence of the channel\_label field.

subsampling\_type: This field indicates the subsampling type as specified by Table XX4.

subsampling\_location: This field indicates the subsampling sample location as specified by Table XX4.

channel\_label: The human readable description of the channel.

Table XX4 - Channel subsampling and sample position

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Value of subsampling\_type | Value of subsampling\_location | Channel subsampling | Position of the centre of the top-left chroma sample relative to the centre of the top-left luma sample in units of luma samples | |
| Horizontal (x) | Vertical (y) |
| 0 | 0 or 1 or 2 or 3 or 4 or 5 | none (4:4:4) | 0 | 0 |
| 1 | 0 or 2 or 4 | subsampled by a factor 2 horizontally (4:2:2) | 0 | 0 |
| 1 or 3 or 5 | 0.5 |
| 2 | 0 | subsampled both horizontally and vertically by a factor 2 (4:2:0) | 0 | 0.5 |
| 1 | 0.5 | 0.5 |
| 2 | 0 | 0 |
| 3 | 0.5 | 0 |
| 4 | 0 | 1 |
| 5 | 0.5 | 1 |
| 3 | 0 or 2 or 4 | subsampled by a factor 4 horizontally (4:1:1) | 0 | 0 |
| 1 or 3 or 5 | 1.5 |
| 4 | 0 or 1 | subsampled by a factor 2 vertically (4:4:0) | 0 | 0.5 |
| 2 or 3 | 0 |
| 4 or 5 | 1 |
| Other values | | Reserved | | |

Change the syntax in subclause 6.5.8.2 to:

aligned(8) class AuxiliaryTypeProperty

extends ItemFullProperty('auxC', version = 0, flags) {

utf8string aux\_type;

unsigned int(8) aux\_subtype[];

// until the end of the box, the semantics depend on the aux\_type value

}

add the following in subclause 6.6.2.5.3 of DAM1 after channel\_id:

channel\_id, when present, shall not be equal to 0 (Unused).

The regions indicated by all instances of channel\_id and packed\_channel\_id not equal to 0 present in the current cfen image item, are used to generate a derived image based on the values of the indicated channel\_id and packed\_channel\_id and any associated colour information.

If is\_packed\_flag is equal to 1, then the picture is first partitioned into (num\_cols\_minus1+1 ) + ( num\_rows\_minus1+1 ) regions as follows:

The width and height of each region is computed as follows:

RegionWidthInLumaSamples = image\_width / num\_cols\_minus1

RegionHeightInLumaSamples = image\_height / num\_rows\_minus1

A region R[j][k] for any value of j in the range of 0 to num\_rows\_minus1, inclusive, and k in the range of 0 to num\_cols\_minus1, inclusive, is associated with the luma area starting from the horizontal and vertical coordinates OrigX[j][k] and OrigY[j][k] that are computed as follows:

OrigX[j][k] = (k \* RegionWidthInLumaSamples) + (hor\_guard\_band\_mul2\*2)

OrigY[j][k] = (j \* RegionHeightInLumaSamples) + (ver\_guard\_band\_mul2\*2)

and end at horizontal and vertical coordinates EndX[j][k] and EndY[j][k] that are computed as follows:

EndX[j][k]= ((k+1)\* RegionWidthInLumaSamples) - (hor\_guard\_band\_mul2\*2+1)

EndX[j][k]= ((j+1)\* RegionHeightInLumaSamples) - (ver\_guard\_band\_mul2\*2+1)

Only the luma samples associated with any indicated regions, as specified above, are considered for the generation of a derived video signal and all other luma and chroma samples are ignored.

Change the sentence in subclause 6.8.11 Image pyramid Entity Group from (DAM1):

When the flag tile\_info\_present\_flag is not set, the tile information of a layer of the image pyramid is derived depending on the image item as described in Tables 2 to 4 below.

To:

When the flag tile\_info\_present\_flag is not set, the tile information of a layer of the image pyramid is derived depending on the image item as described in Tables 2 to 5 below.

Add the following Table 5 in 6.8.11 Image pyramid Entity Group after Table 4 (DAM1):

**Table 5 — Tile information based on Tiled image item 'tili'**

|  |  |
| --- | --- |
| **ImagePyramidEntityGroup tile information** | **Tiled image item 'tili'** |
| *tileWidth* | tile\_width |
| *tileHeight* | tile\_height |
| *tileColumns* | ceil(ispe.image\_width/tile\_width) |
| *tileRows* | ceil(ispe.image\_height/tile\_height) |

Add the following new subclause after subclause 6.11:

## 6.12 Tiled image items

**6.12.1 General**

A tiled image item is constructed of uniform, independently coded tiles arranged in rows, columns, and optionally extra dimensions, to form a rectangular image or n-dimensional hyperrectangle. The tiles are identical in size, format, coding, and makeup and may be compressed or uncompressed.

**6.12.2 Tiled image item**

An image item of type ‘tili’ is a tiled image, with each tile coded independently from other tiles. Input tiles may be stored either in separate external files or in a contiguous range of addressing space to support byte range addressing and retrieval of individual tiles with a single read.

NOTE 1 The image coding method is defined by a writer using a valid image codec 4CC.

NOTE 2 As opposed to a ‘grid’ image item, where the declaration and addressing of tiles occurs in the file-scoped MetaBox, a ‘tili’ image item with contiguous range of addressing space has a single declaration parameter in the file-scoped MetaBox and an associated addressing table, with offsets and extents for each tile, stored with the image tiles, typically in a media data box. This has the advantage that the required file ranges of the addressing table, which can be large for terapixel images, can also be loaded on-demand.

The tiled image item (‘tili’) shall be associated with TiledImageConfigurationBox (‘tilC’) and ImageSpatialExtentsProperty which carries the width and height of the overall tiled image. All the necessary item properties of the tiled image item shall be present in the ItemPropertyBox and is associated to the tiles of the tiled image item through the TilePropertyAssociationBox.

The tiled image item (‘tili’) allows for storing YUV imagery as a single coded tile object. Conversely, each band in a tile can be stored separately, allowing for direct access to each component in a tile separately. This results in a 3D arrangement of tiles within the image. This is useful for multi and hyperspectral imagery.

A tiled image item (‘tili’) can be included within an ImagePyramidEntityGroup. To create a full pyramid, progressively binned tiled image items may be included. When building a multi-resolution pyramid, different image coding methods can be used for each layer.

The location of input tiles to the tiled image item is identified by the corresponding DataEntryTiledItemBox in the DataReferenceBox which is mapped to the tiled image item through data\_reference\_index in the ItemLocationBox.

The DataEntryTiledItemBox shall contain URLs to the input tiles stored in separate external files or shall contain the size and offset to the TiledImageOffsetTable to support byte range addressing and retrieval of individual tiles.

If the input tiles are stored in separate external files, then each external file shall contain only one input tile as an image item and be a HEIF compliant file. The external files shall not contain any entity grouping. The input tile in an external file may contain item properties associated with them. The handler\_type of the tiled image item shall be equal to the handler\_type of the input tile in the external file.

When the overall image dimensions are not an even multiple of the image tile size, the rows are padded on the right to complete the last tile in each row of tiles, and the columns are padded on the bottom to complete the last tile in each column of tiles. The width and height parameters in the ImageSpatialExtentsProperty are set to the size of the image containing valid image content, effectively achieving a crop of the padded boundary area.

**6.12.3 Tiled image configuration**

**6.12.3.1 Definition**

Box type: 'tilC'

Property type: Descriptive item property

Container: ItemPropertyContainerBox

Mandatory (per item): Yes, for image items of type 'tili'

Quantity (per item): One

The TiledImageConfigurationBox specifies parameters associated with a tiled image item (‘tili’) (see Section 6.12). These parameters include the tile resolution, and the image item type used to code and store individual tile content. Configuration information also includes the number and size of additional dimensions when coding n-dimensional hyperrectangles. This includes support for the coding of multi and hyperspectral imagery where each band in a tili tile region is separately retrievable.

The TiledImageConfigurationBox consists of TileItemPropertyAssociationBox which associates tiles to item properties in the ItemPropertiesBox.

**6.12.3.2 Syntax**

aligned(8) class TileItemPropertyAssociationBox

extends FullBox('tipa', version, flags) {

unsigned int(8) association\_count;

for (j=0; j<association\_count; j++) {

bit(1) essential;

if (flags & 1)

unsigned int(15) property\_index;

else

unsigned int(7) property\_index;

}

}

aligned(8) class TiledImageConfigurationBox

extends ItemFullProperty('tilC', version=0, flags) {

unsigned int(32) tile\_width;

unsigned int(32) tile\_height;

unsigned int(8) number\_of\_extra\_dimensions;

for (int i=0; i<number\_of\_extra\_dimensions; i++) {

unsigned int(32) dimension\_size[i];

}

if (DataEntryTiledItemBox.external\_tiles\_urls==0) {

unsigned int(32) tile\_item\_type;

TileItemPropertyAssociationBox tile\_association[];

}

}

**6.12.3.3 Semantics**

tile\_width, tile\_height shall be set to the size of a single tile width and height. All tiles have the same size. Tiles at the right or bottom border of the overall image may include padding when the tile width and or height are not integer multiples of the overall tili item width or height. In this case, the ImageSpatialExtentsProperty is set to the boundary of the true image width and height to achieve a crop of the padded area.

tile\_item\_type specifies the image item type used for all the individual tile images. In a tili item, each tile is coded separately so it can be extracted and decoded independently. tile\_item\_type shall be set to a valid four-character code for a coded image item (e.g., ‘hvc1’ for h265 compression, ‘j2k1’ for JPEG2000, or ‘unci’ for uncompressed). When required by the image item type, all necessary image properties shall be associated using the TileItemPropertyAssociationBox. Certain codecs (jpg, etc.) may not require any configuration properties in such a case the association\_count in the TileItemPropertyAssociationBox is set to 0.

number\_of\_extra\_dimensions specifies the number of extra dimensions if the image resembles a (number\_of\_extra\_dimensions+2)-dimensional hyperrectangle. For a 2D image, number\_of\_extra\_dimensions shall be 0.

For a single layer 2D image (e.g., YUV or monochrome), tiles are indexed with a uniform ordering of the form [y][x], where [x] is the width parameter, and [y] is the height parameter. For a multi-dimensional image where components are stored separately, the tiles are indexed with a uniform ordering of the form [z][y][x], where [z] is the number of coded dimensions parameter.

dimension\_size[i] specifies the size of dimension i+2 of the n-dimensional hyperrectangle. Note that the size of the first two dimensions are the image\_width and image\_height specified in the ImageSpatialExtentsProperty of the tili item.

association\_count indicates the number of item properties associated with the tiles of the tile image item.

essential when set to 1 indicates that the associated property is essential to the tiles of the tiled image item, otherwise it is non-essential

property\_index is either 0 indicating that no property is associated (the essential indicator shall also be 0) or is the 1-based index (counting all boxes, including FreeSpace boxes) of the associated property box in the ItemPropertyContainerBox contained in the ItemPropertiesBox. property\_index shall not be greater than the number of boxes contained in the ItemPropertyContainerBox.

( & 1)value

* + 1. **Tiled image item data**

The payload of a tiled image item (‘tili’) consists of tiles of the item when the external\_tiles\_urls in the associated DataEntryTiledItemBox is set to 0.

* + 1. **Data entry tiled item box**

#### 6.12.5.1 Definition

Box Type: 'deti'  
Container: DataReferenceBox  
Mandatory: No.  
Quantity: Zero or more.

The DataEntryTiledItemBox identifies the location of each tile of a tiled image item ('tili'). The DataEntryTiledItemBox either contains URLs to the input tiles stored in separate external files or contains the size and offset of the TiledImageOffsetTable to support byte range addressing and retrieval of individual tiles.

The TiledImageOffsetTable contains offset pointers and size information for each tile in the item. The TiledImageOffsetTable is stored in the MediaDataBox of the file.

#### 6.12.5.2 Syntax

aligned(8) class TiledImageOffsetTable{

for (int i=0; i < NumTiles; i++) {

unsigned int(DataEntryTiledItemBox.offset\_field\_length) tile\_start\_offset[i];

//not present if size\_field\_length ==0

unsigned int(DataEntryTiledItemBox.size\_field\_length) tile\_size[i];

}

aligned(8) class DataEntryTiledItemBox (bit(24) flags)  
 extends DataEntryBaseBox('deti', flags)   
{

switch (flags & 0x03) {

case 0:

offset\_field\_length = 32;

break;

case 1:

offset\_field\_length = 40;

break;

case 2:

offset\_field\_length = 48;

break;

case 3:

offset\_field\_length = 64;

break;

}

switch ((flags>>2) & 0x03) {

case 0:

size\_field\_length = 0;

break;

case 1:

size\_field\_length = 24;

break;

case 2:

size\_field\_length = 32;

break;

case 3:

size\_field\_length = 64;

break;

}

sequential\_order = ((flags>>4) & 0x01);

switch ((flags>>5) & 0x03) {

case 0:

input\_items\_size\_index = 8;

break;

case 1:

input\_items\_size\_index = 16;

break;

case 2:

input\_items\_size\_index = 32;

break;

case 3:

input\_items\_size\_index = 64;

break;

}

external\_tiles\_urls = ((flags>>7) & 0x01);

unsigned int(input\_items\_size\_index) no\_of\_input\_items;

if(external\_tiles\_urls){

unsigned int(64) tileIDstart;

utf8string baseurl;

utf8string urlextension;

utf8string tileitemrequesttemplate;

}

else{

unsigned int(DataEntryTiledItemBox.offset\_field\_length) tile\_offset\_table\_start\_offset;

unsigned int(DataEntryTiledItemBox.size\_field\_length)

tile\_offset\_table\_size;

}  
}

#### 6.12.5.3 Semantics

offset\_field\_length defines the number of bits used to store the offset to the TiledImageOffsetTable and the offset to the image data of a specific tile in the TiledImageOffsetTable.

size\_field\_length defines the number of bits used to store the length of the TiledImageOffsetTable and the length of the image data of a specific tile in the TiledImageOffsetTable.

sequential\_order when true, indicates that the compressed image tile data is stored consecutively in sequential order.

input\_items\_size\_index specifies the size of the parameters no\_of\_input\_items in bytes. With value 0 indicating size is of 1 byte up to the value 3 indicating the size to be 8 bytes.

external\_tile\_urls when true, indicates that the tiles are stored in external files indicated by the URLs.

The parameter no\_of\_input\_items in DataEntryTiledItemBox shall be equal to:

TileColumns = (ImageSpatialExtentsProperty.image\_width + TiledImageConfigurationBox.tile\_width-1)/ TiledImageConfigurationBox.tile\_width;

TileRows = (ImageSpatialExtentsProperty.image\_height + TiledImageConfigurationBox.tile\_height-1)/ TiledImageConfigurationBox.tile\_height;

no\_of\_input\_items = TileColumns \* TileRows

for (i=0; i< TiledImageConfigurationBox.number\_of\_extra\_dimensions; i++) {

no\_of\_input\_items = no\_of\_input\_items \* TiledImageConfigurationBox.dimension\_size[i];

}

tileIDstart indicates the ID of the first tile in the associated tile image item. The tile IDs start with tileIDstart value until NumTiles + tileIDstart with increments of one in the order it is mapped in the tile image item (row-major order).

baseurl contains the base URL for the tiles.

urlextension contains URL extensions which is used in URL

tileitemrequesttemplate contains the template which is used in URL construction

|  |  |
| --- | --- |
| **base URL** | http://cdn.example.com/pictures/134532/image/ |
| **URL extension** | Representation1 |
| Tile Item Request template | $tileID$.heif |
| Tile ID Start | 1000 |
| **URL** | http://cdn.example.com/pictures/134532/image/Representation1/1000.heif |

In the example above, assuming that the first tile is selected, the URL constructed results in http://cdn.example.com/pictures/134532/image/Representation1/1000.heif

The URL is constructed by concatenating the baseurl and the urlextension and the ‘/’ character followed by the value of the tileitemrequesttemplate, where the tileID is replaced by the actual value of the tile starting from the value given in tileIDstart

[Ed. Note]: the template may include parameters for both tile ID and directory ID, this is to be studied further

tile\_offset\_table\_start\_offset points to the start of the TiledImageOffsetTable. The position is given relative to the referenced data in the ItemLocationBox associated to the image item.

tile\_offset\_table\_size indicates the size in bytes of the TiledImageOffsetTable.

tile\_start\_offset[i] points to the start of the coded data of a tile. The position is given relative to the referenced data in the ItemLocationBox associated to the image item. If a specific tile is empty and does not contain image content, the tile is not coded and the tile\_start\_offset[i] entry shall be set to 0. This situation may occur when an image is generated on a canvas and certain portions of the overall image only contain canvas with no image pixels. Readers shall interpret a tile\_start\_offset[i] value equal to 0 as an empty tile with no media content.

tile\_size[i] (if present) indicates the number of bytes of the coded tile bitstream.

The number of tile offsets stored in the table (NumTiles) is computed by

TileColumns = ceil (ImageSpatialExtentsProperty.image\_width/ TiledImageConfigurationBox.tile\_width);

TileRows = ceil (ImageSpatialExtentsProperty.image\_height/ TiledImageConfigurationBox.tile\_height);

NumTiles = TileColumns \* TileRows

for (i=0; i< TiledImageConfigurationBox.number\_of\_extra\_dimensions; i++) {

NumTiles = NumTiles \* TiledImageConfigurationBox.dimension\_size[i];

}

TileColumns and TileRows are the number of tiles in a row within the overall image and the number of tiles in a column within the overall image. image\_width and image\_height are the dimensions of the entire image as specified in the associated ImageSpatialExtentsProperty item property. number\_of\_extra\_dimensions and dimension\_size [] is defined in the TiledImage

ConfigurationBox property associated with the tili item. NumTiles represents the number of tiles in the entire tiled image item.

The entries in the offset table are ordered in row-major sequence. For a 2D image with a single coded layer, they are indexed as [y][x], where:

x = tile column

y = tile row

For a 3D tiled image item, they are indexed as [z][y][x], where:

x = tile column

y = tile row

z = depth coordinate

For a general n-dimensional hyperrectangle, the tiles are indexed as [zn-1] [zn-2] ...[z3] [z2][y][x], where zi are the n-2 extra dimensions.

x is the inner most looping variable, followed by y, and then z2 to zn-1.

The coded tile data may be stored in the file in arbitrary order, resulting in the tile\_start\_offset entries not necessarily being in increasing address order.

When size\_field\_length==0, the tile\_size[i] variables are not present, and the decoder infers them from the difference between the tile\_start\_offset entries. For the case where tiles are stored in sequential order (flags & 0x10 == 0x10), the tile\_size[i] is computed as

tile\_start\_offset[i+1] - tile\_start\_offset[i], except for the last tile, which extends until the end of the data. If the tiles are not stored in sequential order, the decoder first sorts the tile\_start\_offset entries before computing the size from the offset differences. In this case, the decoder cannot read the offset table on-demand. For on-demand applications, the tile sizes should be included. When multiple tiles contain the same content, the tile\_start\_offset entries for these tiles may point to the same data block. In this case, sequential ordering is not used.

NOTE In some cases, it is possible to encounter unspecified tiled regions. In such cases the rendering of unspecified regions is left to reader/player.

Add the following paragraph at the beginning of subclause 10.2.5.1:

This brand may be used when the item type of a coded image item allows multiple coded pictures within the same coded image item, such as coded pictures representing different spatial scalability layers of the same picture, but the coded image item contains only one coded picture that is intra coded. This brand implies no semantics or reader requirements, when the item type of a coded image item does not allow multiple coded pictures in the same coded image item.

Change the sentence in subclause 10.2.5.1 from:

This brand enables file players to identify and decode HEIF files containing coded image items that only contain one picture, and that picture is intra coded.

to:

This brand enables file players to identify and decode HEIF files containing coded image items that only contain one coded picture and that coded picture is intra coded.

Change the sentence in subclause 10.2.5.2 from:

A file having the '1pic' brand in the compatible\_brands array of the FileTypeBox shall contain coded image items that only contain one picture, and that picture is intra coded.

to:

A file having the '1pic' brand in the compatible\_brands array of the FileTypeBox shall contain coded image items that only contain one coded picture, and that coded picture is intra coded.

Add the following new subclauses after subclause 11.3.3:

#### 11.3.4 Groups of regions

##### 11.3.4.1 Union of regions entity group

A union of regions entity group ('unrg') indicates the union of all the regions represented by one or more region items.

Each entity\_id value in the entity group shall refer to a region item.

All the region items in the union of regions entity group shall be associated with the same image item, inside which the regions are defined, using an item reference of type 'cdsc' from each region item to the same image item.

If unique IDs are used:

* the union of regions entity group may also be associated with the image item inside which the regions are defined using an item reference of type 'cdsc' from the union of regions entity group to the image item,
* an annotation may be associated with the union of regions entity group by associating:
  + an item property, using the ItemPropertyAssociationBox;
  + a metadata item, using an item reference of type 'cdsc' from the metadata item to the union of regions entity group.
  + an image item or another entity group, using an item reference of type 'eroi' from the union of regions entity group to the image item or the other entity group.

##### 11.3.4.2 Compound region entity group

A compound region entity group ('corg') associates one main region item with one or more region items. It indicates an inclusion relationship between a main object covered by regions of a main entity and other objects covered by regions described by one or more other entities, the main object logically including the other objects.

NOTE 1 For example, a compound region entity group can be used to associate a main region corresponding to a body with regions corresponding to body parts (e.g., the head, legs or arms of the body) to indicate that the body is logically including the body parts.

The entities in a compound region entity group shall be region items.

The number of entities in a compound region entity group shall be at least 2. The first entity\_id value shall indicate the main region item. It indicates the region covering the main object that is logically including the objects covered by the regions described by the second and following entity\_ids.

This inclusion relationship does not convey information at the geometry level. A main region signaled as including other regions by a compound region entity group may or may not geometrically include the other regions.

NOTE 2 For example, the main region item corresponding to the first entity\_id value can represent a bounding box or a region encompassing partially the regions described by the region items corresponding to the second and following entity\_id values.

All the region items in the compound region entity group shall be associated with the same image item, inside which the regions are defined, using an item reference of type 'cdsc' from each region item to the same image item.

If unique IDs are used:

* the compound region entity group may also be associated with the image item, inside which the regions are defined, using an item reference of type 'cdsc' from the compound region entity group to the image item,
* an annotation may be associated with the compound region entity group by associating:
  + an item property, using the ItemPropertyAssociationBox;
  + a metadata item, using an item reference of type 'cdsc' from the metadata item to the compound region entity group.
  + an image item or another entity group, using an item reference of type 'eroi' from the union of regions entity group to the image item or the other entity group.

Add the following to the end of clause A.2.1:

When an ExifDataBlock is compressed using the deflate() algorithm defined in IETF RFC 1951, and the resulting untimed compressed Exif metadata is stored as a metadata item, the item\_type value shall be set to 'dExf'.

Replace the contents of subclause J.2 with:

A file with a single coded image item, Exif metadata and T.35 metadata could be structured as follows:

FileTypeBox 'ftyp': major-brand='heic', compatible-brands='heic'

MetaBox 'meta': (container)

HandlerBox 'hdlr': 'pict'

PrimaryItemBox 'pitm': item\_ID=1;

ItemInfoBox 'iinf': entry\_count=3

1) 'infe': item\_ID=1, item\_type='hvc1';

2) 'infe': item\_ID=2, item\_type='Exif';

3) 'infe': item\_ID=3, item\_type='it35'; // ITU-T T.35

ItemLocationBox 'iloc': item\_count=3

item\_ID=1, extent\_count=1, extent\_offset=X, extent\_length=Y;

item\_ID=2, extent\_count=1, extent\_offset=P, extent\_length=Q;

item\_ID=3, extent\_count=1, extent\_offset=R, extent\_length=S;

ItemReferenceBox 'iref':

referenceType='cdsc', from\_item\_ID=2, ref\_count=1, to\_item\_ID=1;

referenceType='cdsc', from\_item\_ID=3, ref\_count=1, to\_item\_ID=1;

ItemPropertiesBox 'iprp':

ItemPropertyContainerBox 'ipco':

'hvcC'

'ispe'

ItemPropertyAssociation 'ipma': entry\_count=1

1) item\_ID=1, association\_count=2

essential=1, property\_index=1;

essential=0, property\_index=2;

MediaDataBox 'mdat' or 'idat':

HEVC Image (at file offset X, with length Y)

Exif data block (at file offset P, with length Q)

T.35 data block (at file offset R, with length S)

Add a new subclause J.7:

### J.7 Single image in a low-overhead image file

A file with a single HEVC encoded image item could be structured as follows:

FileTypeBox 'ftyp':

major\_brand='mif3'  
minor\_version='heic'  
compatible\_brands=''

MinimizedImageBox 'mini': (container)

version=0

explicit\_codec\_types\_flag=0  
float\_flag=0  
full\_range\_flag=1  
alpha\_flag=0  
explicit\_cicp\_flag=0  
hdr\_flag=0  
icc\_flag=0  
exif\_flag=0  
xmp\_flag=0

chroma\_subsampling=3  
orientation\_minus1=0

large\_dimensions\_flag=0  
width\_minus1=W  
height\_minus1=H

high\_bit\_depth\_flag=0

large\_codec\_config\_flag=0  
large\_item\_data\_flag=1

main\_item\_codec\_config\_size=C,  
main\_item\_data\_size\_minus1=D,  
main\_item\_codec\_config=HEVCDecoderConfigurationRecord (with length C)  
main\_item\_data=HEVC Image Data (with length D+1)

Rename clause L.4.1

### L.4.1 'mif1'-compliant VVC image and image collection brands

Add new subclause L.4.3

### L.4.3 'mif3'-compliant VVC image and image collection brand

#### L.4.3.1 Requirements on files

Files shall include 'mif3' as the major\_brand and the brand 'vvi3' as the minor\_version in the FileTypeBox and conform to the specifications in ‎O.2.1 of this document ('mif3' structural brand).

The 'vvi3' brand defines the coded image item type to be 'vvc1' as defined in L.2.2.1.2 and the alpha\_item\_codec\_config (if present), the gainmap\_item\_codec\_config (if present) and the main\_item\_codec\_config to have the format defined by CompactVvcDecoderConfigurationRecord specified in L.4.3.3 of this document.

Files that include 'mif3' as the major\_brand and the brand 'vvi3' as the minor\_version in the FileTypeBox shall comply with the specifications in ‎L.2 and shall additionally be constrained as follows:

* explicit\_codec\_types\_flag in the MinimizedImageBox shall be equal to 0,
* float\_flag in the MinimizedImageBox shall be 0,
* If gainmap\_item\_data\_size in the MinimizedImageBox is not 0, gainmap\_float\_flag in the MinimizedImageBox shall be 0,
* If alpha\_flag in the MinimizedImageBox is equal to 1, alpha\_item\_data\_size in the MinimizedImageBox shall not be equal to 0,
* The data of each codec configuration property shall use the syntax defined in subclause L.4.3.3.2 of this document,
* Each coded image item shall contain a single IDR picture per layer,
* When multi\_layer\_flag is equal to 0, the data of each coded image item shall consist of only one NAL unit excluding the length and NAL unit header fields, which are inferred by the reader as specified in subclause L.4.3.2 of this document.,
* The equivalent file as specified in L.4.3.1 shall conform to the 'vvic' brand.

#### L.4.3.2 Requirements on readers

The requirements on readers specified in ‎L.4.1.3 and ‎O.2.1 of this document ('mif3' structural brand) shall be supported.

As a response to 'mif3' as the major\_brand and the brand 'vvi3' as the minor\_version in the FileTypeBox, the readers shall treat the file as if an equivalent file were created containing FileTypeBox with 'vvic' as the equivalent major\_brand and the equivalent MetaBox and MediaDataBox derived with the following additional operations in addition to those specified in subclause O.4:

* infer infe\_type to be equal to 'vvc1',
* infer codec\_config\_type to be equal to 'vvcC',
* expand the data of each codec configuration property from the compact syntax to the full structure as defined in subclause L.4.3.3.4 of this document,
* when multi\_layer\_flag is equal to 0, expand the data of the respective image item so that the NAL unit length is inferred from the item length, and so that the NAL unit header fields nal\_unit\_type, nuh\_temporal\_id\_plus1 and nuh\_layer\_id are set to 8 (IDR\_N\_LP), 1 and 0, respectively.

#### L.4.3.3 Compact VVC decoder configuration

##### L.4.3.3.1 Definition

This subclause specifies the compact decoder configuration information for ISO/IEC 23090-3 video content. The compact decoder configuration provides essential parameters that are relevant for still images.

##### L.4.3.3.2 Syntax

aligned(8) class CompactVvcDecoderConfigurationRecord {

   unsigned int(1) multi\_layer\_flag;

   unsigned int(2) lengthSizeMinusOne;

unsigned int(5) num\_nal\_units;

for (i=0; i< num\_nal\_units; i++) {

unsigned int(8) nal\_unit\_length[ i ];

bit(8\* nal\_unit\_length) nal\_unit[ i ];

}

}

**L.4.3.3.3 Semantics**

multi\_layer\_flag: When equal to 0, the data of the coded image item shall consist of only one NAL unit that excludes the length and NAL unit header fields and has nal\_unit\_type, nuh\_temporal\_id\_plus1 and nuh\_layer\_id equal to 8 (IDR\_N\_LP), 1 and 0, respectively. When multi\_layer\_flag is equal to 1, the data of the coded image item may include any number of NAL units, which include the NAL unit length and NAL unit header fields.

num\_nal\_units indicates the number of NAL units, included in the configuration record for the referenced CVS.

nal\_unit\_length[i] indicates the length in bytes of the ith NAL unit.

nal\_unit[i] contains a NAL unit as specified in ISO/IEC 23090-3.

The semantics of other parameters are the same as for VvcDecoderConfigurationRecord as defined in ISO/IEC 14496-15.

##### L.4.3.3.4 Equivalence with the VVC decoder configuration

CompactVvcDecoderConfigurationRecord shall be considered equivalent to VvcDecoderConfigurationRecord as defined in ISO/IEC 14496-15 with the following fields:

* ptl\_present\_flag is set to 1:
* num\_sublayers is set to 1,
* constant\_frame\_rate is set to 1,
* if the codec configuration property is associated with the main image item:
  + chroma\_format\_idc is set to the value of the chroma\_subsampling field from the MinimizedImageBox,
* if the codec configuration property is associated with the alpha auxiliary image item:
  + chroma\_format\_idc is set to 0,
* if the codec configuration property is associated with the main image item or with the alpha auxiliary image item:
  + bit\_depth\_minus8 is set to 0 if the value of the high\_bit\_depth\_flag field from the MinimizedImageBox is 0, or to the value of the bit\_depth\_minus9 field from the MinimizedImageBox plus 1,
  + max\_picture\_width is set to the value plus 1 of the width\_minus1 field from the MinimizedImageBox,
  + max\_picture\_height is set to the value plus 1 of the height\_minus1 field from the MinimizedImageBox,
* if the codec configuration property is associated with the gain map image item:
  + bit\_depth\_minus8 is set to 0 if the value of the gainmap\_high\_bit\_depth\_flag field from the MinimizedImageBox is 0, or to the value of the gainmap\_bit\_depth\_minus9 field from the MinimizedImageBox plus 1,
  + max\_picture\_width is set to the value plus 1 of the gainmap\_width\_minus1 field from the MinimizedImageBox,
  + max\_picture\_height is set to the value plus 1 of the gainmap\_height\_minus1 field from the MinimizedImageBox,
* avg\_frame\_rate is set to 0,
* The native\_ptl is determined as follows:
  + When multi\_layer\_flag is set to 0,
    - native\_ptl parameters are inferred from the profile\_tier\_level( 1, sps\_max\_sublayers\_minus1 ) structure as defined in the seq\_parameter\_set\_rbsp( ) as per ISO/IEC 23090-3 of the SPS NAL unit
  + Otherwise, when multi\_layer\_flag is set to 1,
    - the ols\_idx is set equal to opi\_ols\_idx of the OPI NAL unit, if it is present in the corresponding item data of the MinimizedImageBox or is set equal to lowest OLS index that contains the largest number of layers among all OLSs specified by the VPS NAL unit and the largest number of output layers among the OLSs with the largest number of layers.
    - The profile indication general\_profile\_idc shall indicate a profile to which the output layer set identified by ols\_idx conforms.
    - The tier indication general\_tier\_flag shall indicate a tier equal to or greater than the highest tier indicated in all the profile\_tier\_level( ) syntax structures (in all the parameter sets) to which the output layer set identified by ols\_idx in this configuration record conforms.
    - A bit at a particular bit index in general\_constraint\_info shall be equal to 0 when the bit of that same particular bit index is equal to 0 in any general\_constraints\_info( ) syntax structure among all the profile\_tier\_level( ) syntax structures in all the parameter sets referred to by the referenced CVSs.
    - The level indication general\_level\_idc shall indicate a level of capability equal to or greater than the highest level in all the profile\_tier\_level( ) syntax structures (in all the parameter sets) to which the operating point identified by ols\_idx and num\_sublayers in this configuration record conforms.
    - All other parameters are set based on the information present in the VPS NAL unit.
* if multi\_layer\_flag is set to 0, the num\_nal\_units is >= 2, with at least the SPS NAL unit and PPS NAL unit (as defined in ISO/IEC 23090-3) present in the CompactVvcDecoderConfigurationRecord.
* if num\_nal\_units is greater than 1 each i-th NAL unit nal\_unit[i] is processed as follows:
  + num\_of\_arrays is initialized to 0
  + The NAL unit type of nal\_unit[i] (nal\_unit\_type as defined in ISO/IEC 23090-3) is determined
    - There is NAL unit array with NAL\_unit\_type set to nal\_unit\_type of the nal\_unit[i]
    - If the nal\_unit[i] is the first NAL unit with this NAL unit type, num\_arrays is incremented by 1
    - num\_nalus is set to 1 if it is the first NAL unit of the array, otherwise num\_nalus is incremented by 1.
    - A new set of nal\_unit\_length and nal\_unit is added to the array as follows
      * nal\_unit\_length is set to nal\_unit\_length[i] of the CompactVvcDecoderConfigurationRecord,
      * nal\_unit is set to nal\_unit[i] of the CompactVvcDecoderConfigurationRecord.
* the array\_completeness is set equal to 1 in VvcDecoderConfigurationRecord.
* the other parameters are carried over as is, and repeated if needed.

##### L.4.4.3.5 Equivalence with the TargetOLSproperty

When multi\_layer\_flag is set to 1 in CompactVvcDecoderConfigurationRecord, the CompactVvcDecoderConfigurationRecord shall be considered equivalent to TargetOLSproperty as defined in ISO/IEC 23008-12 with the following fields:

* the target\_ols\_idx in TargetOLSproperty is set equal to opi\_ols\_idx of the OPI NAL unit, if it is present in the corresponding item data of the MinimizedImageBox or is set equal to lowest OLS index that contains the largest number of layers among all OLSs specified by the VPS NAL unit and the largest number of output layers among the OLSs with the largest number of layers.

##### L.4.4.3.6 Equivalence with the LayerSelectorProperty

When multi\_layer\_flag is set to 1 in CompactVvcDecoderConfigurationRecord, the CompactVvcDecoderConfigurationRecord shall be considered equivalent to LayerSelectorProperty as defined in ISO/IEC 23008-12 with the following fields:

* the layer\_id in LayerSelectorProperty is set equal to the output layer having the greater nuh\_layer\_id value in the target OLS.

##### L.4.4.3.7 Equivalence with the VvcOperatingPointsInformationProperty

When multi\_layer\_flag is set to 1 in CompactVvcDecoderConfigurationRecord, the CompactVvcDecoderConfigurationRecord shall be considered equivalent to VvcOperatingPointsRecord as defined in ISO/IEC 23008-12.

NOTE The reader builds the VvcOperatingPointsRecord using the VPS NAL unit present in the CompactVvcDecoderConfigurationRecord.

Add new Annex O:

**Annex O**(normative)

**Low-overhead image file format**

### O.1 General

The low-overhead image file format provides a more compact representation of the image file format for specific use-cases. This format is designed for small and simple files where the traditional use of the MetaBox would result in significant overhead relative to the size of the image and/or metadata payloads. In this format, the top-level MetaBox and MediaDataBox are replaced by a MinimizedImageBox, which logically maintains the presence of the MetaBox and MediaDataBox by representing its contents.

When a parser encounters a MinimizedImageBox, it shall expand it to a MetaBox and a MediaDataBox as described in Clause O.4, at which point the specifications of Clause 6 apply.

When a brand specified in Clause O.2 is the major brand or among the compatible brands of a file, either explicitly or implicitly, the requirements specified in this annex shall be applied.

### O.2 Brands

#### O.2.1 'mif3' structural brand

##### O.2.1.1 Requirements on files

Files containing the brand 'mif3' as the major brand or in the compatible brands array of the FileTypeBox shall conform to the constraints defined in this subclause.

When the 'mif3' brand is the major brand or present among the compatible brands of the FileTypeBox, the file may be identified by the MIME type defined in Annex P. When the 'mif3' brand is the major brand, the defined file extension and MIME type defined in Annex P should be used.

The boxes listed in Table XX1 are required in a file under the 'mif3' brand. The Version column in Table XX1 lists the versions of the boxes allowed by this brand. Other versions of the boxes shall not be present. Other file-level boxes shall not be present.

**Table XX1 — Required boxes in a file under the** 'mif3' **brand**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Hierarchy of boxes** | | | **Version** | **Box description** |
| ftyp |  |  | - | file type and compatibility |
| mini |  |  | 0 | metadata and image data |

When the 'mif3' brand is present as the major brand of the FileTypeBox, the minor version of the FileTypeBox shall be 0 or a brand that is either structurally compatible with the 'mif3' brand, such as a codec brand complying with the 'mif3' structural brand, or a brand to which the file conforms after the equivalent MetaBox and MediaDataBox have been transformed from MinimizedImageBox as specified in Clause O.4.

##### O.2.1.2 Requirements on readers

Support for the boxes listed in Table XX2 is required under the 'mif3' brand in addition to support for the boxes required to be supported under the 'mif1' brand. The Version column in Table XX2 specifies the versions of the boxes that shall be supported by the readers of the 'mif3' brand.

**Table XX2 — Boxes to be supported under the 'mif3' brand  
in addition to those required for readers of 'mif1' brand**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Hierarchy of boxes** | | | **Version** | **Box description** |
| mini |  |  | 0 | metadata and image data |

A file containing a MinimizedImageBox shall be treated as if:

* it contained the 'mif1' brand in the compatible\_brands array in the FileTypeBox,
* it contained the equivalent MetaBox as described in Clause O.4,
* the major\_brand of the FileTypeBox was the equivalent brand specified by the brand provided in the minor\_version, if the major\_brand is 'mif3' and the minor\_version is not 0,
* it contained the 'tmap' brand in the compatible\_brands array in the FileTypeBox if gainmap\_flag as defined in clause O.3.3 is set to 1 in the MinimizedImageBox,
* it contained the equivalent MediaDataBox as described in Clause O.4.

### O.3 Minimized Image Box

#### O.3.1 Definition

|  |  |
| --- | --- |
| Box type: | 'mini' |
| Container: | File |
| Mandatory: | No |
| Quantity: | Zero or one |

The MinimizedImageBox provides a more compact representation of the MetaBox and MediaDataBox and is a substantial part of the low-overhead image file format.

When MinimizedImageBox is present in a file, the 'mif3' brand or a derived brand that implies the 'mif3' brand shall be the major brand or present among the compatible brands in the FileTypeBox.

#### O.3.2 Syntax

aligned(8) class MinimizedImageBox extends Box('mini') {

bit(2) version = 0;

// flags

bit(1) explicit\_codec\_types\_flag;

bit(1) float\_flag;

bit(1) full\_range\_flag;

bit(1) alpha\_flag;

bit(1) explicit\_cicp\_flag;

bit(1) hdr\_flag;

bit(1) icc\_flag;

bit(1) exif\_flag;

bit(1) xmp\_flag;

bit(2) chroma\_subsampling;

bit(3) orientation\_minus1;

// Spatial extents

bit(1) large\_dimensions\_flag;

unsigned int(large\_dimensions\_flag ? 15 : 7) width\_minus1;

unsigned int(large\_dimensions\_flag ? 15 : 7) height\_minus1;

// Pixel information

if (chroma\_subsampling == 1 || chroma\_subsampling == 2)

bit(1) chroma\_is\_horizontally\_centered;

if (chroma\_subsampling == 1)

bit(1) chroma\_is\_vertically\_centered;

if (float\_flag)

bit(2) bit\_depth\_log2\_minus4;

else {

bit(1) high\_bit\_depth\_flag;

if (high\_bit\_depth\_flag)

bit(3) bit\_depth\_minus9;

}

if (alpha\_flag)

bit(1) alpha\_is\_premultiplied;

// Colour properties

if (explicit\_cicp\_flag) {

bit(8) colour\_primaries;

bit(8) transfer\_characteristics;

bit(8) matrix\_coefficients;

}

else {

colour\_primaries = icc\_flag ? 2 : 1;

transfer\_characteristics = icc\_flag ? 2 : 13;

matrix\_coefficients = chroma\_subsampling == 0 ? 2 : 6;

}

if (explicit\_codec\_types\_flag) {

bit(32) infe\_type;

bit(32) codec\_config\_type;

} else {

// codec information is defined by the brand in the minor\_version field of FileTypeBox

}

// High Dynamic Range properties

if (hdr\_flag) {

bit(1) gainmap\_flag;

if (gainmap\_flag) {

bit(1) gainmap\_dimension\_same\_as\_main\_item\_flag;

if(gainmap\_dimension\_same\_as\_main\_item\_flag){

unsigned int() gainmap\_width\_minus1 = width\_minus1;

unsigned int() gainmap\_height\_minus1 = height\_minus1;

}

else{

unsigned int(large\_dimensions\_flag ? 15 : 7) gainmap\_width\_minus1;

unsigned int(large\_dimensions\_flag ? 15 : 7) gainmap\_height\_minus1;

}

bit(8) gainmap\_matrix\_coefficients;

bit(1) gainmap\_full\_range\_flag;

bit(2) gainmap\_chroma\_subsampling;

if (gainmap\_chroma\_subsampling == 1 || gainmap\_chroma\_subsampling == 2)

bit(1) gainmap\_chroma\_is\_horizontally\_centered;

if (gainmap\_chroma\_subsampling == 1)

bit(1) gainmap\_chroma\_is\_vertically\_centered;

bit(1) gainmap\_float\_flag;

if (gainmap\_float\_flag)

bit(2) gainmap\_bit\_depth\_log2\_minus4;

else {

bit(1) gainmap\_high\_bit\_depth\_flag;

if (gainmap\_high\_bit\_depth\_flag)

bit(3) gainmap\_bit\_depth\_minus9;

}

bit(1) tmap\_icc\_flag;

bit(1) tmap\_explicit\_cicp\_flag;

if (tmap\_explicit\_cicp\_flag) {

bit(8) tmap\_colour\_primaries;

bit(8) tmap\_transfer\_characteristics;

bit(8) tmap\_matrix\_coefficients;

bit(1) tmap\_full\_range\_flag;

}

else {

tmap\_colour\_primaries = 1;

tmap\_transfer\_characteristics = 13;

tmap\_matrix\_coefficients = 6;

tmap\_full\_range\_flag = 1;

}

}

bit(1) clli\_flag;

bit(1) mdcv\_flag;

bit(1) cclv\_flag;

bit(1) amve\_flag;

bit(1) reve\_flag;

bit(1) ndwt\_flag;

if (clli\_flag)

ContentLightLevel clli;

if (mdcv\_flag)

MasteringDisplayColourVolume mdcv;

if (cclv\_flag)

ContentColourVolume cclv;

if (amve\_flag)

AmbientViewingEnvironment amve;

if (reve\_flag)

ReferenceViewingEnvironment reve;

if (ndwt\_flag)

NominalDiffuseWhite ndwt;

if (gainmap\_flag) {

bit(1) tmap\_clli\_flag;

bit(1) tmap\_mdcv\_flag;

bit(1) tmap\_cclv\_flag;

bit(1) tmap\_amve\_flag;

bit(1) tmap\_reve\_flag;

bit(1) tmap\_ndwt\_flag;

if (tmap\_clli\_flag)

ContentLightLevel tmap\_clli;

if (tmap\_mdcv\_flag)

MasteringDisplayColourVolume tmap\_mdcv;

if (tmap\_cclv\_flag)

ContentColourVolume tmap\_cclv;

if (tmap\_amve\_flag)

AmbientViewingEnvironment tmap\_amve;

if (tmap\_reve\_flag)

ReferenceViewingEnvironment tmap\_reve;

if (tmap\_ndwt\_flag)

NominalDiffuseWhite tmap\_ndwt;

}

}

// Chunk sizes

if (icc\_flag || exif\_flag || xmp\_flag || (hdr\_flag && gainmap\_flag))

bit(1) large\_metadata\_flag;

bit(1) large\_codec\_config\_flag;

bit(1) large\_item\_data\_flag;

if (icc\_flag)

unsigned int(large\_metadata\_flag ? 20 : 10) icc\_data\_size\_minus1;

if (hdr\_flag && gainmap\_flag && tmap\_icc\_flag)

unsigned int(large\_metadata\_flag ? 20 : 10) tmap\_icc\_data\_size\_minus1;

if (hdr\_flag && gainmap\_flag)

unsigned int(large\_metadata\_flag ? 20 : 10) gainmap\_metadata\_size;

if (hdr\_flag && gainmap\_flag)

unsigned int(large\_item\_data\_flag ? 28 : 15) gainmap\_item\_data\_size;

if (hdr\_flag && gainmap\_flag && gainmap\_item\_data\_size > 0)

unsigned int(large\_codec\_config\_flag ? 12 : 3) gainmap\_item\_codec\_config\_size;

unsigned int(large\_codec\_config\_flag ? 12 : 3) main\_item\_codec\_config\_size;

unsigned int(large\_item\_data\_flag ? 28 : 15) main\_item\_data\_size\_minus1;

if (alpha\_flag)

unsigned int(large\_item\_data\_flag ? 28 : 15) alpha\_item\_data\_size;

if (alpha\_flag && alpha\_item\_data\_size > 0)

unsigned int(large\_codec\_config\_flag ? 12 : 3) alpha\_item\_codec\_config\_size;

if (exif\_flag || xmp\_flag)

unsigned int(1) exif\_xmp\_compressed\_flag;

if (exif\_flag)

unsigned int(few\_metadata\_bytes\_flag ? 10 : 20) exif\_data\_size\_minus1;

if (xmp\_flag)

unsigned int(few\_metadata\_bytes\_flag ? 10 : 20) xmp\_data\_size\_minus1;

trailing\_bits(); // bit padding till byte alignment

// Chunks

unsigned int(8) main\_item\_codec\_config[main\_item\_codec\_config\_size];

unsigned int(8) alpha\_item\_codec\_config[]; // non-parsed variable

if (alpha\_flag && alpha\_item\_data\_size > 0){

if(alpha\_item\_codec\_config\_size == 0){

alpha\_item\_codec\_config\_size = main\_item\_codec\_config\_size;

alpha\_item\_codec\_config = main\_item\_codec\_config;

}

else{

unsigned int(8)

alpha\_item\_explicit\_codec\_config[alpha\_item\_codec\_config\_size];

alpha\_item\_codec\_config = alpha\_item\_explicit\_codec\_config;

}

}

unsigned int(8) gainmap\_item\_codec\_config[]; // non-parsed variable

if (hdr\_flag && gainmap\_flag && gainmap\_item\_data\_size > 0){

if(gainmap\_item\_codec\_config\_size == 0){

gainmap\_item\_codec\_config\_size = main\_item\_codec\_config\_size;

gainmap\_item\_codec\_config = main\_item\_codec\_config;

}

else{

unsigned int(8)

gainmap\_item\_explicit\_codec\_config[gainmap\_item\_codec\_config\_size];

gainmap\_item\_codec\_config = gainmap\_item\_explicit\_codec\_config;

}

}

if (icc\_flag)

unsigned int(8) icc\_data[icc\_data\_size\_minus1 + 1];

if (hdr\_flag && gainmap\_flag && tmap\_icc\_flag)

unsigned int(8) tmap\_icc\_data[tmap\_icc\_data\_size\_minus1 + 1];

if (hdr\_flag && gainmap\_flag && gainmap\_metadata\_size > 0)

unsigned int(8) gainmap\_metadata[gainmap\_metadata\_size];

if (alpha\_flag && alpha\_item\_data\_size > 0)

unsigned int(8) alpha\_item\_data[alpha\_item\_data\_size];

if (hdr\_flag && gainmap\_flag && gainmap\_item\_data\_size > 0)

unsigned int(8) gainmap\_item\_data[gainmap\_item\_data\_size];

unsigned int(8) main\_item\_data[main\_item\_data\_size\_minus1 + 1];

if (exif\_flag)

unsigned int(8) exif\_data[exif\_data\_size\_minus1 + 1];

if (xmp\_flag)

unsigned int(8) xmp\_data[xmp\_data\_size\_minus1 + 1];

}

#### O.3.3 Semantics

version: specifies the version of the MinimizedImageBox. The version shall be set to 0 in this version of this document.

explicit\_codec\_types\_flag: 0 specifies that infe\_type and codec\_config\_type, are not present and are inferred as specified for the brand carried in minor\_version of the FileTypeBox. 1 specifies that infe\_type and codec\_config\_type are present. explicit\_codec\_types\_flag shall be equal to 1, when the major\_brand in the FileTypeBox is neither ‘mif3’ nor a brand derived from ‘mif3’ or minor\_version does not specify a brand that specified the inference of infe\_type and codec\_config\_type. When the major\_brand in the FileTypeBox is ‘mif3’ or a brand derived from ‘mif3’, minor\_version specifies a brand, and explicit\_codec\_types\_flag is equal to 1, infe\_type and codec\_config\_type that are present in the MinimizedImageBox are in effect instead of those that would be inferred based on the brand carried in minor\_version.

float\_flag: specifies the format of the pixel values of the reconstructed main and alpha image items as the component\_format 0 and 1 values, as specified in PixelInformationProperty with px\_flags&1!=0 in clause 6.5.6.

full\_range\_flag: is a binary value representing the VideoFullRangeFlag as defined in Rec. ITU-T H.273 | ISO/IEC 23091-2 for the main image when float\_flag is 0. Values 0 and 1 are reserved when float\_flag is 1.

alpha\_flag: 0 specifies that the image is opaque. Otherwise, the image has an alpha layer, whether the codec has native translucency support, or an alpha auxiliary image item is used.

explicit\_cicp\_flag: equal to 0 indicates the values of ColourPrimaries, TransferCharacteristics and MatrixCoefficients, as defined in Rec. ITU-T H.273 | ISO/IEC 23091-2, are set to the values defined in Table XX2. When the value is equal to 1 it indicates that these values are signaled explicitly, as defined in Table XX2.

Table XX2 - Values of CICP fields associated with the main image

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **value of explicit\_cicp\_flag** | **value of icc\_flag** | **value of chroma\_subsampling** | **Main image on-screen colors** | **ColourPrimaries is set to** | **TransferCharacteristics is set to** | **MatrixCoefficients is set to** |
| 0 | 0 | 0 (4:0:0) | Grayscale | 1 | 13 | 2 |
| 1, 2 or 3 (4:2:X or 4:4:4) | sRGB | 6 |
| 1 | 0 (4:0:0) | Defined by the ICC profile associated with the main image | 2 | 2 | 2 |
| 1, 2 or 3 (4:2:X or 4:4:4) | 6 |
| 1 | any | 0 (4:0:0) | Defined by the CICP triplet and/or the ICC profile associated with the main image | colour\_primaries | transfer\_characteristics | 2 |
| 1, 2 or 3 (4:2:X or 4:4:4) | matrix\_coefficients |

hdr\_flag: 0 specifies that the image is SDR (Standard Dynamic Range) and has no associated HDR-related signaling (High Dynamic Range). 1 specifies that the image is either SDR with an SDR-to-HDR gain map, or HDR with an optional HDR-to-SDR gain map.

icc\_flag: equal to 1 indicates that the main image is associated with an ICC profile as defined in ISO 15076-1 or ICC.1[13].

exif\_flag: equal to 1 indicates the presence of Exif metadata.

xmp\_flag: equal to 1 indicates the presence of XMP metadata.

chroma\_subsampling: Specifies the number of colour channels and the subsampling of the chroma channels, as defined in Table XX1, of the main image.

Table XX1 - Chroma subsampling

|  |  |  |
| --- | --- | --- |
| **Value of chroma\_subsampling** | **Number of color channels** | **Chroma subsampling** |
| 0 | 1 (for example grayscale) | monochrome (4:0:0) |
| 1 | 3 (for example YUV) | subsampled both horizontally and vertically by a factor 2 (4:2:0) |
| 2 | 3 (for example YUV) | subsampled by a factor 2 horizontally (4:2:2) |
| 3 | 3 (for example RGB or YUV) | no (4:4:4) |

orientation\_minus1: plus 1 specifies the Exif orientation value as defined for the Orientation tag in JEITA CP-3451.

large\_dimensions\_flag: if set to 0, the length of the fields signaled among width\_minus1, height\_minus1, gainmap\_width\_minus1 and gainmap\_height\_minus1 is 7 bits; otherwise, it is 15 bits.

width\_minus1: plus 1 specifies the width of the reconstructed image in pixels.

height\_minus1: plus 1 specifies the height of the reconstructed image in pixels.

chroma\_is\_horizontally\_centered: 0 specifies that the chroma samples of the main image are co-located horizontally with the luma samples of the main image, otherwise they are horizontally centered between the luma samples of the main image. 0 unless chroma\_subsampling is 1 or 2.

chroma\_is\_vertically\_centered: 0 specifies that the chroma samples of the main image are co-located vertically with the luma samples of the main image, otherwise they are vertically centered between the luma samples of the main image. 0 unless chroma\_subsampling is 1.

bit\_depth\_log2\_minus4: specifies the format of floating-point numbers used for the pixel values of the reconstructed main and alpha image items. The values 0, 1, and 2 respectively correspond to the bits\_per\_channel values 16, 32 and 64, as specified in PixelInformationProperty in clause 6.5.6. Other values are reserved. When float\_flag is set to 0, the value is undefined.

high\_bit\_depth\_flag: 0 specifies that the number of bits per channel for the pixel values of the reconstructed main and alpha image items, as specified in PixelInformationProperty in clause 6.5.6, is 8. Otherwise, bit\_depth\_minus9 is signaled. When float\_flag is set to 1, the value is undefined.

bit\_depth\_minus9: specifies the number of bits, minus nine, per channel for the pixel values of the reconstructed main and alpha image items, as specified in PixelInformationProperty in clause 6.5.6. When high\_bit\_depth\_flag is set to 0 or float\_flag is set to 1, the value is undefined.

alpha\_is\_premultiplied: when set to 1 specifies that the color channels are pre-multiplied by the alpha channel, otherwise the color channels are not pre-multiplied. Ignored if alpha\_flag is 0.

colour\_primaries: carries a ColourPrimaries value as defined in Rec. ITU-T H.273 | ISO/IEC 23091-2 for the main image.

transfer\_characteristics: carries a TransferCharacteristics value as defined in Rec. ITU-T H.273 | ISO/IEC 23091-2 for the main image.

matrix\_coefficients: carries a MatrixCoefficients value as defined in Rec. ITU-T H.273 | ISO/IEC 23091-2 for the main image.

infe\_type: carries the coded image item type. Corresponds to the item\_type field of the version 2 of the ItemInfoEntry box. Defined by the brand carried by the minor\_version of the FileTypeBox if explicit\_codec\_types\_flag is 0.

codec\_config\_type: carries the codec configuration property box type. Defined by the brand carried by the minor\_version of the FileTypeBox if explicit\_codec\_types\_flag is 0.

gainmap\_flag: 0 specifies that the file has no tone-mapped image and no associated HDR-related ISO 21496-1 gain map. Otherwise, the file contains a tone-mapped image and is associated with a gain map, whether the codec has native gain map support, or a separate gain map image item is used. 0 if hdr\_flag is 0.

gainmap\_dimension\_same\_as\_main\_item\_flag: 0 specifies that the values of gainmap\_width\_minus1 and gainmap\_height\_minus1 are signaled. 1 specifies that the values of gainmap\_width\_minus1 and gainmap\_height\_minus1 are copied from width\_minus1 and height\_minus1, respectively, instead of being signaled.

gainmap\_width\_minus1: carries the width minus one of the gain map image in pixels.

gainmap\_height\_minus1: carries the height minus one of the gain map image in pixels.

gainmap\_matrix\_coefficients: carries a MatrixCoefficients value as defined in Rec. ITU-T H.273 | ISO/IEC 23091-2 for the gain map image.

gainmap\_full\_range\_flag: is a binary value representing the VideoFullRangeFlag as defined in Rec. ITU-T H.273 | ISO/IEC 23091-2 for the gain map image when float\_flag is 0. Values 0 and 1 are reserved when float\_flag is 1.

gainmap\_chroma\_subsampling: Specifies the number of colour channels and the subsampling of the chroma channels, as defined in Table XX1, of the gain map image.

gainmap\_chroma\_is\_horizontally\_centered: 0 specifies that the chroma samples of the gain map image are co-located horizontally with the luma samples of the gain map image, otherwise they are horizontally centered between the luma samples of the gain map image. Ignored unless gainmap\_chroma\_subsampling is 1 or 2.

gainmap\_chroma\_is\_vertically\_centered: 0 specifies that the chroma samples of the gain map image are co-located vertically with the luma samples of the gain map image, otherwise they are vertically centered between the luma samples of the gain map image. Ignored unless gainmap\_chroma\_subsampling is 1.

gainmap\_float\_flag: specifies the format of the pixel values of the reconstructed gain map image item as the component\_format values, as specified in PixelInformationProperty with px\_flags&1!=0 in clause 6.5.6.

gainmap\_bit\_depth\_log2\_minus4: specifies the format of floating-point numbers used for the pixel values of the reconstructed gain map image item. The values 0, 1, and 2 respectively correspond to the bits\_per\_channel values 16, 32 and 64, as specified in PixelInformationProperty in clause 6.5.6. Other values are reserved. When float\_flag is set to 0, the value is undefined.

gainmap\_high\_bit\_depth\_flag: 0 specifies that the number of bits per channel for the pixel values of the reconstructed gain map image item, as specified in PixelInformationProperty in clause 6.5.6, is 8. Otherwise gainmap\_bit\_depth\_minus9 is signaled. When gainmap\_float\_flag is set to 1, the value is undefined.

gainmap\_bit\_depth\_minus9: specifies the number of bits, minus nine, per channel for the pixel values of the reconstructed gain map image item, as specified in PixelInformationProperty in clause 6.5.6. When high\_bit\_depth\_flag is set to 0 or float\_flag is set to 1, the value is undefined.

tmap\_icc\_flag: if 1, specifies that the tone-mapped image is associated with an ICC profile as defined in ISO 15076-1 or ICC.1[23]. 0 if gainmap\_flag is 0.

tmap\_explicit\_cicp\_flag: 0 specifies sRGB on-screen colors as the values of ColourPrimaries, TransferCharacteristics and MatrixCoefficients, as defined in Rec. ITU-T H.273 | ISO/IEC 23091-2, associated with the tone-mapped image, set to the values defined in Table XX3. Otherwise, these values are signaled explicitly, as defined in Table XX3.

Table XX3 - Values of CICP fields associated with the tone-mapped image

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **value of tmap\_explicit\_cicp\_flag** | **Tone-mapped image on-screen colors** | **ColourPrimaries is set to** | **TransferCharacteristics is set to** | **MatrixCoefficients is set to** |
| 0 | sRGB | 1 | 13 | 6 |
| 1 | Defined by the CICP triplet and/or the ICC  profile associated with the tone-mapped image | tmap\_colour\_primaries | tmap\_transfer\_characteristics | tmap\_matrix\_coefficients |

tmap\_colour\_primaries: carries a ColourPrimaries value as defined in Rec. ITU-T H.273 | ISO/IEC 23091-2 for the tone-mapped image.

tmap\_transfer\_characteristics: carries a TransferCharacteristics value as defined in Rec. ITU-T H.273 | ISO/IEC 23091-2 for the tone-mapped image.

tmap\_matrix\_coefficients: carries a MatrixCoefficients value as defined in Rec. ITU-T H.273 | ISO/IEC 23091-2 for the tone-mapped image.

tmap\_full\_range\_flag: is a binary value representing the VideoFullRangeFlag as defined in Rec. ITU-T H.273 | ISO/IEC 23091-2 for the tone-mapped image when float\_flag is 0. Values 0 and 1 are reserved when float\_flag is 1. Set to 1 if tmap\_explicit\_cicp\_flag is 0.

clli\_flag: 1 specifies that there is signaling for ContentLightLevel attached to the main image. Otherwise, no such signaling is present. 0 if hdr\_flag is 0.

mdcv\_flag: 1 specifies that there is signaling for MasteringDisplayColourVolume attached to the main image. Otherwise, no such signaling is present. 0 if hdr\_flag is 0.

cclv\_flag: 1 specifies that there is signaling for ContentColourVolume attached to the main image. Otherwise, no such signaling is present. 0 if hdr\_flag is 0.

amve\_flag: 1 specifies that there is signaling for AmbientViewingEnvironment attached to the main image. Otherwise, no such signaling is present. 0 if hdr\_flag is 0.

reve\_flag: 1 specifies that there is signaling for ReferenceViewingEnvironment attached to the main image. Otherwise, no such signaling is present. 0 if hdr\_flag is 0.

ndwt\_flag: 1 specifies that there is signaling for NominalDiffuseWhite attached to the main image. Otherwise, no such signaling is present. 0 if hdr\_flag is 0.

clli: The box body of the ContentLightLevelBox as defined in ISO/IEC 14496-12 attached to the main image. Only present if clli\_flag is 1.

mdcv: The box body of the MasteringDisplayColourVolumeBox as defined in ISO/IEC 14496-12 attached to the main image. Only present if mdcv\_flag is 1.

cclv: The box body of the ContentColourVolumeBox as defined in ISO/IEC 14496-12 attached to the main image. Only present if cclv\_flag is 1.

amve: The box body of the AmbientViewingEnvironmentBox as defined in ISO/IEC 14496-12 attached to the main image. Only present if amve\_flag is 1.

reve: The box body of the ReferenceViewingEnvironmentBox attached to the main image. Only present if reve\_flag is 1.

ndwt: The box body of the NominalDiffuseWhiteBox attached to the main image. Only present if ndwt\_flag is 1.

tmap\_clli\_flag: 1 specifies that there is signaling for ContentLightLevel attached to the tone-mapped image. Otherwise, no such signaling is present. 0 if gainmap\_flag is 0.

tmap\_mdcv\_flag: 1 specifies that there is signaling for MasteringDisplayColourVolume attached to the tone-mapped image. Otherwise, no such signaling is present. 0 if gainmap\_flag is 0.

tmap\_cclv\_flag: 1 specifies that there is signaling for ContentColourVolume attached to the tone-mapped image. Otherwise, no such signaling is present. 0 if gainmap\_flag is 0.

tmap\_amve\_flag: 1 specifies that there is signaling for AmbientViewingEnvironment attached to the tone-mapped image. Otherwise, no such signaling is present. 0 if gainmap\_flag is 0.

tmap\_reve\_flag: 1 specifies that there is signaling for ReferenceViewingEnvironment attached to the tone-mapped image. Otherwise, no such signaling is present. 0 if gainmap\_flag is 0.

tmap\_ndwt\_flag: 1 specifies that there is signaling for NominalDiffuseWhite attached to the tone-mapped image. Otherwise, no such signaling is present. 0 if gainmap\_flag is 0.

tmap\_clli: The box body of the ContentLightLevelBox as defined in ISO/IEC 14496-12 attached to the tone-mapped image. Only present if tmap\_clli\_flag is 1.

tmap\_mdcv: The box body of the MasteringDisplayColourVolumeBox as defined in ISO/IEC 14496-12 attached to the tone-mapped image. Only present if tmap\_mdcv\_flagis 1.

tmap\_cclv: The box body of the ContentColourVolumeBox as defined in ISO/IEC 14496-12 attached to the tone-mapped image. Only present if tmap\_cclv\_flag is 1.

tmap\_amve: The box body of the AmbientViewingEnvironmentBox as defined in ISO/IEC 14496-12 attached to the tone-mapped image. Only present if tmap\_amve\_flag is 1.

tmap\_reve: The box body of the ReferenceViewingEnvironmentBox attached to the tone-mapped image. Only present if tmap\_reve\_flag is 1.

tmap\_ndwt: The box body of the NominalDiffuseWhiteBox attached to the tone-mapped image. Only present if tmap\_ndwt\_flag is 1.

large\_metadata\_flag: 0 specifies that the length of the signaled fields among icc\_data\_size\_minus1, tmap\_icc\_data\_size\_minus1, gainmap\_metadata\_size, exif\_data\_size\_minus1 and xmp\_data\_size\_minus1 is 10 bits, otherwise 20 bits. Undefined unless one of these fields is signaled.

large\_codec\_config\_flag: 0 specifies that the length of the signaled fields among gainmap\_item\_codec\_config\_size, main\_item\_codec\_config\_size and alpha\_item\_codec\_config\_size is 3 bits, otherwise 12 bits.

large\_item\_data\_flag: 0 specifies that the length of the signaled fields among gainmap\_item\_data\_size, main\_item\_data\_size\_minus1 and alpha\_item\_data\_size is 15 bits, otherwise 28 bits.

icc\_data\_size\_minus1: carries the size minus one in bytes of the ICC profile as defined in ISO 15076-1 or ICC.1[23], associated with the main image. Undefined if icc\_flag is 0.

tmap\_icc\_data\_size\_minus1: carries the size minus one in bytes of the ICC profile as defined in ISO 15076-1 or ICC.1[23], associated with the tone-mapped image. Undefined if tmap\_icc\_flag is 0.

gainmap\_metadata\_size: carries the size of the gain map metadata. 0 if gainmap\_flag is 0.

gainmap\_item\_data\_size: carries the size of the coded sample data for the HDR-related gain map image item in bytes. If gainmap\_flag is set to 1, a size of 0 is reserved for future use. 0 if gainmap\_flag is 0.

gainmap\_item\_codec\_config\_size: carries the size of the codec configuration for the gain map auxiliary image item in bytes. The value 0 is replaced by the value of main\_item\_codec\_config\_size and specifies that the gainmap\_item\_codec\_config payload is identical to the main\_item\_codec\_config payload. 0 if gainmap\_item\_data\_size is 0.

main\_item\_codec\_config\_size: carries the size of the codec configuration for the main image item in bytes.

main\_item\_data\_size\_minus1: carries the size minus one of the coded sample data for the main image item in bytes.

alpha\_item\_data\_size: carries the size of the coded sample data for the alpha auxiliary image item in bytes. If alpha\_flag is set to 1, the value 0 specifies that the codec has native translucency support and that the alpha samples are coded alongside the color samples in the main\_item\_data chunk. 0 if alpha\_flag is 0.

alpha\_item\_codec\_config\_size: carries the size of the codec configuration for the alpha auxiliary image item in bytes. The value 0 is replaced by the value of main\_item\_codec\_config\_size and specifies that the alpha\_item\_codec\_config payload is identical to the main\_item\_codec\_config payload. 0 if alpha\_item\_data\_size is 0.

exif\_xmp\_compressed\_flag: when set to 1, indicates that the Exif metadata and XMP metadata payloads, if present, are each compressed using the deflate() algorithm as defined in IETF RFC 1951. When set to 0, the Exif metadata and XMP metadata payloads are not compressed.

exif\_data\_size\_minus1: specifies the size minus one of the Exif metadata in bytes if exif\_xmp\_compressed\_flag equals 0. It specifies the size minus one of the compressed Exif metadata in bytes if exif\_xmp\_compressed\_flag equals 1. -1 if exif\_flag is 0.

xmp\_data\_size\_minus1: specifies the size minus one of the XMP metadata in bytes if exif\_xmp\_compressed\_flag equals 0. It specifies the size minus one of the compressed XMP metadata in bytes if exif\_xmp\_compressed\_flag equals 1. -1 if xmp\_flag is 0.

trailing\_bits: padding bits to ensure payloads are 8-bit aligned. Shall be 0.

main\_item\_codec\_config: carries the main image item codec configuration data. When main\_item\_codec\_config\_size is 0, main\_item\_codec\_config is not present.

alpha\_item\_codec\_config: carries the optional alpha image codec configuration data. When alpha\_item\_codec\_config\_size is 0, alpha\_item\_codec\_config is not present.

gainmap\_item\_codec\_config: carries the HDR-related gain map image item codec configuration data. When gainmap\_item\_codec\_config\_size is 0, gainmap\_item\_codec\_config is not present.

icc\_data: carries the ICC profile data of the main image as defined in ISO 15076-1 or ICC.1[23]. When icc\_flag is 0, icc\_data is not present.

tmap\_icc\_data: carries the ICC profile data of the optional HDR-related tone-mapped image as defined in ISO 15076-1 or ICC.1[23]. When tmap\_icc\_flag is 0, tmap\_icc\_data is not present.

gainmap\_metadata: Gain map metadata as defined by the GainMapMetadata struct in ISO 21496-1. Not present if gainmap\_metadata\_size is 0.

alpha\_item\_data: carries the coded sample data of the optional alpha image. When alpha\_item\_data\_size is 0, alpha\_item\_data is not present.

gainmap\_item\_data: carries the coded sample data of the optional gain map image. When gainmap\_item\_data\_size is 0, gainmap\_item\_data is not present.

main\_item\_data: carries the coded sample data of the main image.

exif\_data: specifies the optional Exif metadata corresponding to the ExifDataBlock specified in clause A.2.1 if exif\_xmp\_compressed\_flag equals 0. Specifies the optional compressed Exif metadata corresponding to the ExifDataBlock specified in clause A.2.1 if exif\_xmp\_compressed\_flag equals 1. When exif\_flag is set to 0, exif\_data is not present.

xmp\_data: specifies the optional XMP metadata if exif\_xmp\_compressed\_flag equals 0. Specifies the optional compressed XMP metadata if exif\_xmp\_compressed\_flag equals 1. When xmp\_flag is set to 0, xmp\_data is not present.

### O.4 Equivalence with MetaBox

A MinimizedImageBox has a one-to-one mapping to a MetaBox and MediaDataBox. Readers shall treat a MinimizedImageBox as if it were the equivalent MetaBox and MediaDataBox that are transformed from MinimizedImageBox as specified in this subclause. If readers do not explicitly reconstruct the equivalent MetaBox and MediaDataBox in memory upon parsing, the output image shall be the same as if they did.

NOTE 1 When a reader encounters the MinimizedImageBox, it can create the equivalent MetaBox and MediaDataBox in memory and populate its contents based on the parsed contents of the MinimizedImageBox to reuse regular MetaBox handling implementation.

File writers can either write an image file based on the MetaBox or write an image file based on the MinimizedImageBox.

NOTE 2 File writers can write an image file based on the MinimizedImageBox when the MetaBox would result in considerable overhead compared to the image data payload. File writers can strip metadata such as Exif and XMP, entirely or to its strict minimum, to avoid bloating the MinimizedImageBox.

The equivalent MetaBox has its sub-boxes described by the following subclauses.

#### O.4.1 HandlerBox

The equivalent MetaBox shall have a HandlerBox with handler\_type equal to 'pict'.

#### O.4.2 PrimaryItemBox

The equivalent MetaBox shall have a PrimaryItemBox with item\_ID set to 1.

#### O.4.3 ItemInfoBox

The equivalent MetaBox shall have an ItemInfoBox containing the following entries:

* ItemInfoEntry of version 2 with flags set to 0, item\_ID set to 1 and item\_type set to infe\_type. All other fields are set to null or 0 as appropriate.
* If alpha\_item\_data\_size is not 0, ItemInfoEntry of version 2 with flags set to 1, item\_ID set to 2 and item\_type set to infe\_type. All other fields are set to null or 0 as appropriate.
* If gainmap\_flag is 1, ItemInfoEntry of version 2 with flags set to 0, item\_ID set to 3 and item\_type set to 'tmap'. All other fields are set to null or 0 as appropriate.
* If gainmap\_item\_data\_size is not 0, ItemInfoEntry of version 2 with flags set to 1, item\_ID set to 4 and item\_type set to infe\_type. All other fields are set to null or 0 as appropriate.
* If exif\_flag is 1, and exif\_xmp\_compressed\_flag is 0, ItemInfoEntry of version 2 with flags set to 1, item\_ID set to 6 and item\_type set to 'Exif'. If exif\_flag is 1 and exif\_xmp\_compressed\_flag is 1, ItemInfoEntry of version 2 with flags set to 1, item\_ID set to 6 and item\_type set to 'dExf'” to indicate that the EXIF metadata is compressed and needs to be decoded before interpreted. All other fields are set to null or 0 as appropriate.
* If xmp\_flag is 1, and exif\_xmp\_compressed\_flag is 0, ItemInfoEntry of version 2 with flags set to 1, item\_ID set to 7 and item\_type set to 'mime' and content\_type set to 'application/rdf+xml'. If exif\_xmp\_compressed\_flag is 1, the content\_encoding is set to “deflate” to indicate that the XMP metadata is compressed and needs to be decoded before being interpreted. All other fields are set to null or 0 as appropriate.

#### O.4.4 ItemReferenceBox

The ItemReferenceBox is populated with the following entries:

* If ItemInfoBox has an entry for item\_ID 2: Item type reference with referenceType set to 'auxl', from\_item\_ID set to 2, reference\_count set to 1 and to\_item\_ID set to 1.
* If ItemInfoBox has an entry for item\_ID 2 and alpha\_is\_premultiplied is set to 1: Item type reference with referenceType set to 'prem', from\_item\_ID set to 1, reference\_count set to 1 and to\_item\_ID set to 2.
* If gainmap\_flag is 1 and ItemInfoBox has an entry for item\_ID 4: item type reference with referenceType set to 'dimg', from\_item\_ID set to 3, reference\_count set to 2 and the following to\_item\_ID entries: 1, 4.
* If gainmap\_flag is 1 and ItemInfoBox has zero entry for item\_ID 4: item type reference with referenceType set to 'dimg', from\_item\_ID set to 3, reference\_count set to 1 and to\_item\_ID set to 1.
* If ItemInfoBox has an entry for item\_ID 6: item type reference with referenceType set to 'cdsc', from\_item\_ID set to 6, reference\_count set to 1 and to\_item\_ID set to 1.
* If ItemInfoBox has an entry for item\_ID 7: item type reference with referenceType set to 'cdsc', from\_item\_ID set to 7, reference\_count set to 1 and to\_item\_ID set to 1.

If the resulting ItemReferenceBox contains at least one entry, it shall be added to the equivalent MetaBox. An empty ItemReferenceBox shall be ignored.

#### O.4.5 EntityToGroupBox

If gainmap\_flag is 1, the equivalent MetaBox shall have a GroupsListBox containing a single sub-box. That sub-box is an EntityToGroupBox with grouping\_type set to 'altr', version set 0, flags set to 0, group\_id set to 5, num\_entities\_in\_group set to 2 and the following entity\_id entries: 3, 1.

#### O.4.6 ItemPropertiesBox

The equivalent MetaBox shall have an ItemPropertiesBox containing an ItemPropertyContainerBox and an ItemPropertyAssociationBox.

The ItemPropertyContainerBox shall have 32 entries as listed below. Any entry for which the condition is not true is replaced with a FreeSpaceBox.

|  |  |  |
| --- | --- | --- |
| **Entry** | **Condition** | **Contents** |
| 1 | main\_item\_codec\_config\_size is not 0 | Item property with the type set to codec\_config\_type and with contents from main\_item\_codec\_config. |
| 2 | true | ImageSpatialExtentsProperty with image\_width set to width\_minus1+1 and image\_height set to height\_minus1+1. |
| 3 | true | PixelInformationProperty with field values defined in section O.4.7 of this document. |
| 4 | true | ColourInformationBox with colour\_type set to 'nclx' and colour\_primaries, transfer\_characteristics, matrix\_coefficients and full\_range\_flag set to the values from the MinimizedImageBox. |
| 5 | icc\_flag is 1 | ColourInformationBox with the colour\_type set to 'prof' and with ICC\_profile contents being icc\_data. |
| 6 | alpha\_item\_codec\_config\_size is not 0 | Item property with the type set to codec\_config\_type and with contents from alpha\_item\_codec\_config. |
| 7 | alpha\_item\_data\_size is not 0 | AuxiliaryTypeProperty with aux\_type set to 'urn:mpeg:mpegB:cicp:systems:auxiliary:alpha'. |
| 8 | alpha\_item\_data\_size is not 0 | PixelInformationProperty with field values defined in section O.4.7 of this document. |
| 9 | orientation\_minus1 is 2, 4, 5, 6 or 7 | ImageRotation property with angle set to 2, 1, 3, 1, 1, respectively |
| 10 | orientation\_minus1 is 1, 3, 4 or 6 | ImageMirror property with axis set to 1, 0, 0, 1, respectively |
| 11 | clli\_flag is 1 | ContentLightLevelBox with body set to clli |
| 12 | mdcv\_flag is 1 | MasteringDisplayColourVolumeBox with body set to mdcv |
| 13 | cclv\_flag is 1 | ContentColourVolumeBox with body set to cclv |
| 14 | amve\_flag is 1 | AmbientViewingEnvironmentBox with body set to amve |
| 15 | reve\_flag is 1 | ReferenceViewingEnvironmentBox with body set to reve and version and flags set to 0 |
| 16 | ndwt\_flag is 1 | NominalDiffuseWhiteBox with body set to ndwt and version and flags set to 0 |
| 17 | gainmap\_item\_codec\_config\_size is not 0 | Item property with the type set to codec\_config\_type and with contents from gainmap\_item\_codec\_config. |
| 18 | gainmap\_item\_data\_size is not 0 | ImageSpatialExtentsProperty with image\_width set to gainmap\_width\_minus1 + 1 and image\_height set to gainmap\_height\_minus1 + 1. |
| 19 | gainmap\_item\_data\_size is not 0 | PixelInformationProperty with field values defined in section O.4.7 of this document. |
| 20 | gainmap\_item\_data\_size is not 0 | ColourInformationBox with colour\_type set to 'nclx' and colour\_primaries and transfer\_characteristics set to 2, matrix\_coefficients set to gainmap\_matrix\_coefficients and full\_range\_flag set to gainmap\_full\_range\_flag. |
| 21 | gainmap\_flag is 1 | ImageSpatialExtentsProperty equal to entry 2 if orientation\_minus1 is 0, 1, 2, 3 and with image\_width set to height\_minus1 + 1 and image\_height set to width\_minus1 + 1 for any other orientation. |
| 22 | gainmap\_flag is 1 and (tmap\_explicit\_cicp\_flag is 1 or tmap\_icc\_flag is 0) | ColourInformationBox with colour\_type set to 'nclx' and colour\_primaries, transfer\_characteristics, matrix\_coefficients and full\_range\_flag set to tmap\_colour\_primaries, tmap\_transfer\_characteristics, tmap\_matrix\_coefficients and tmap\_full\_range\_flag, respectively. |
| 23 | gainmap\_flag is 1 and tmap\_icc\_flag is 1 | ColourInformationBox with the colour\_type set to 'prof' and with ICC\_profile contents being tmap\_icc\_data. |
| 24 | tmap\_clli\_flag is 1 | ContentLightLevelBox with body set to tmap\_clli |
| 25 | tmap\_mdcv\_flag is 1 | MasteringDisplayColourVolumeBox with body set to tmap\_mdcv |
| 26 | tmap\_cclv\_flag is 1 | ContentColourVolumeBox with body set to tmap\_cclv |
| 27 | tmap\_amve\_flag is 1 | AmbientViewingEnvironmentBox with body set to tmap\_amve |
| 28 | tmap\_reve\_flag is 1 | ReferenceViewingEnvironmentBox with body set to tmap\_reve and version and flags set to 0 |
| 29 | tmap\_ndwt\_flag is 1 | NominalDiffuseWhiteBox with body set to tmap\_ndwt and version and flags set to 0 |
| 30 | alpha\_flag is 1 and alpha\_item\_data\_size is 0 | AlphaInformationProperty with is\_normalised set to 0, is\_premultiplied set to alpha\_is\_premultiplied, opaque\_value set to 2bit\_depth\_log2\_minus4+4 if float\_flag is 1, or to 8 if high\_bit\_depth\_flag is 0, or to bit\_depth\_minus9+9 otherwise, transparent\_value set to 0, and reserved set to 0. |
| 31 | false | Reserved |
| 32 | false | Reserved |

The ItemPropertyAssociationBox shall have the entries below. The order shall be kept to satisfy the constraints in subclause 6.5.1. Any association to a FreeSpaceBox shall be dropped.

* Item 1 shall be associated with ItemPropertyContainerBox entries:
  + 1, essential
  + 2, non-essential
  + 3, non-essential
  + 4, essential
  + 5, essential
  + If alpha\_flag is 1 and alpha\_item\_data\_size is 0,
    - 30, essential
  + If hdr\_flag is 1,
    - 11, non-essential
    - 12, non-essential
    - 13, non-essential
    - 14, non-essential
    - 15, non-essential
    - 16, non-essential
  + 9, essential
  + 10, essential
* If alpha\_item\_data\_size is not 0, item 2 shall be associated with ItemPropertyContainerBox entries:
  + 6, essential
  + 2, non-essential
  + 7, essential
  + 8, non-essential
  + 9, essential
  + 10, essential
* If gainmap\_flag is 1, item 3 shall be associated with ItemPropertyContainerBox entries:
  + 21, non-essential
  + 22, essential
  + 23, essential
  + 24, non-essential
  + 25, non-essential
  + 26, non-essential
  + 27, non-essential
  + 28, non-essential
  + 29, non-essential
* If gainmap\_item\_data\_size is not 0, item 4 shall be associated with ItemPropertyContainerBox entries:
  + 17, essential
  + 18, non-essential
  + 19, non-essential
  + 20, essential
  + 9, essential
  + 10, essential

#### O.4.7 PixelInformationProperty

##### O.4.7.1 Reconstruction

The various PixelInformationProperty boxes associated with the image items in the file are reconstructed given the arguments

* main\_components
* alpha\_flag
* subsampling
* chroma\_is\_horizontally\_centered
* chroma\_is\_vertically\_centered
* float\_flag
* bit\_depth\_log2\_minus4
* high\_bit\_depth\_flag
* bit\_depth\_minus9

Reconstruction happens as follows:

* version set to 0
* px\_flags set to 3
* num\_channels set to:
  + main\_components if alpha\_flag is false
  + main\_components + 1 if alpha\_flag is true
* each bits\_per\_channel entry set to 2bit\_depth\_log2\_minus4+4 if float\_flag is 1, or to 8 if high\_bit\_depth\_flag is 0, or to bit\_depth\_minus9+9 otherwise
* channel\_label\_flag is set to 0 for all entries
* the following entries for channel subsampling and formats
  + the following entry if alpha\_flag is true (skipped if false)
    - channel\_idc set to 1
    - component\_format set to float\_flag,
    - subsampling\_type set to 0
    - subsampling\_location set to 0
  + the following entry if main\_components > 0 (skipped otherwise)
    - channel\_idc set to 0
    - component\_format set to float\_flag,
    - subsampling\_type set to 0
    - subsampling\_location set to 0
  + the following entry, repeated twice, if main\_components > 1 (skipped otherwise)
    - channel\_idc set to 0
    - component\_format set to float\_flag,
    - subsampling\_type set to
      * 2 if subsampling is 1
      * 1 if subsampling is 2
      * 0 if subsampling is 3
    - subsampling\_location set to
      * 1 if chroma\_is\_horizontally\_centered and chroma\_is\_vertically\_centered are both true
      * 3 if chroma\_is\_horizontally\_centered is true and chroma\_is\_vertically\_centered is false
      * 0 if chroma\_is\_horizontally\_centered is false and chroma\_is\_vertically\_centered is true
      * 2 if chroma\_is\_horizontally\_centered and chroma\_is\_vertically\_centered are both false

##### O.4.7.2 Main image PixelInformationProperty

The PixelInformationProperty associated with the main image item is reconstructed as described in section O.4.7.1 with the following arguments:

* main\_components = 1 if chroma\_subsampling is 0, else 3
* alpha\_flag = alpha\_flag is 1 and alpha\_item\_data\_size is 0
* subsampling = chroma\_subsampling
* chroma\_is\_horizontally\_centered = chroma\_is\_horizontally\_centered
* chroma\_is\_vertically\_centered = chroma\_is\_vertically\_centered
* float\_flag = float\_flag
* bit\_depth\_log2\_minus4 = bit\_depth\_log2\_minus4
* high\_bit\_depth\_flag = high\_bit\_depth\_flag
* bit\_depth\_minus9 = bit\_depth\_minus9

##### O.4.7.3 Alpha auxiliary image PixelInformationProperty

If alpha\_item\_data\_size is not 0, the PixelInformationProperty associated with the alpha auxiliary image item is reconstructed as described in section O.4.7.1 with the following arguments:

* main\_components = 0
* alpha\_flag = 1
* subsampling = 0
* chroma\_is\_horizontally\_centered = 0
* chroma\_is\_vertically\_centered = 0
* float\_flag = float\_flag
* bit\_depth\_log2\_minus4 = bit\_depth\_log2\_minus4
* high\_bit\_depth\_flag = high\_bit\_depth\_flag
* bit\_depth\_minus9 = bit\_depth\_minus9

##### O.4.7.4 Gain map image PixelInformationProperty

If gainmap\_flag is not 0, the PixelInformationProperty associated with the gain map image item is reconstructed as described in section O.4.7.1 with the following arguments:

* main\_components = 1 if gainmap\_chroma\_subsampling is 0, else 3,
* alpha\_flag = 0
* subsampling = gainmap\_chroma\_subsampling
* chroma\_is\_horizontally\_centered = gainmap\_chroma\_is\_horizontally\_centered
* chroma\_is\_vertically\_centered = gainmap\_chroma\_is\_vertically\_centered
* float\_flag = gainmap\_float\_flag
* bit\_depth\_log2\_minus4 = gainmap\_bit\_depth\_log2\_minus4
* high\_bit\_depth\_flag = gainmap\_high\_bit\_depth\_flag
* bit\_depth\_minus9 = gainmap\_bit\_depth\_minus9

#### O.4.8 ToneMapImage metadata

If gainmap\_flag is 1, tmap\_item\_data is defined as a data chunk of tmap\_item\_data\_size bytes, being gainmap\_metadata\_size+1 bytes, containing the ToneMapImage metadata as defined in section 6.6.2.4.2, with the following:

* version set to 0
* GainMapMetadata set to gainmap\_metadata

If gainmap\_flag is 0, tmap\_item\_data is defined as an empty chunk and tmap\_item\_data\_size as 0 byte.

#### O.4.9 ItemLocationBox

The equivalent MetaBox shall have an ItemLocationBox of version 1 or version 2 containing the following entries:

* item\_ID 1, with construction\_method set to 0, offset set to mdat\_offset+alpha\_item\_data\_size+tmap\_item\_data\_size+gainmap\_item\_data\_size and length set to main\_item\_data\_size\_minus1+1
* Optional item\_ID 2, with construction\_method set to 0, offset set to mdat\_offset and length set to alpha\_item\_data\_size
* Optional item\_ID 3, with construction\_method set to 0, offset set to mdat\_offset+alpha\_item\_data\_size, and length set to tmap\_item\_data\_size
* Optional item\_ID 4, with construction\_method set to 0, offset set to mdat\_offset+alpha\_item\_data\_size+tmap\_item\_data\_size, and length set to gainmap\_item\_data\_size
* Optional item\_ID 6, with construction\_method set to 0, offset set to mdat\_offset+main\_item\_data\_size\_minus1+1+alpha\_item\_data\_size+tmap\_item\_data\_size+gainmap\_item\_data\_size, and length set to exif\_data\_size\_minus1+1
* Optional item\_ID 7, with construction\_method set to 0, offset set to mdat\_offset+main\_item\_data\_size\_minus1+1+alpha\_item\_data\_size+tmap\_item\_data\_size+gainmap\_item\_data\_size+exif\_data\_size\_minus1+1, and length set to xmp\_data\_size\_minus1+1

with mdat\_offset being the offset in bytes from the beginning of the file till the equivalent MediaDataBox as described in subclause O.4.10, plus the size in bytes of its BoxHeader.

#### O.4.10 MediaDataBox

The equivalent MetaBox shall be immediately followed by an equivalent MediaDataBox containing the non-empty payloads among alpha\_item\_data, tmap\_item\_data, gainmap\_item\_data, main\_item\_data, exif\_data and xmp\_data, concatenated in that order.

The payloads for Exif and XMP are compressed payloads when exif\_xmp\_compressed\_flag=1.

Add new Annex P:

**Annex P**

(normative)

**Low-overhead image file format MIME type registration**

### P.1 General

The file extension and MIME type of a file deriving from the ISO base media file format usually reflect the major brand in the FileTypeBox. When the major brand indicates a brand related to Annex O (low-overhead image file format), the MIME type defined in this annex should be used. When such a brand is a major or compatible brand, this MIME type may also be used.

The registration below is the formal MIME type registration as recorded at IANA.

### P.2 Registration

MIME media type name: image

MIME subtype name: hif2

The semantics of the subtype are as follows:

hif2: High efficiency image file in low-overhead image file

format conforming to the requirements for the 'mif3' brand

using any coding format.

Required parameters: none

Optional parameters:

profiles: Specified by RFC 6381 and its successors

codecs: Specified by RFC 6381 and its successors for files  
 conforming to specifications derived from ISO/IEC  
 14496-12. Note that for ISO-defined (MPEG) video  
 codecs, the format of a list item included in the  
 value of the codecs parameter is specified in  
 ISO/IEC 14496-15.

Encoding considerations: as for video/mp4

Security considerations: See section 4 of RFC 4337 and section 7 of

RFC 6381. This format does not provide integrity or

confidentiality protection and so they are applied externally

when needed. The security considerations of URLs are discussed

in RFC 3986. See also Annex N of ISO/IEC 23008-12.

Interoperability considerations: See implementation in libavif

(https://github.com/AOMediaCodec/libavif).

EDITORS NOTE: URL can be removed depending on IANA preferences.

Published specification: ISO/IEC 23008-12

Applications: Multimedia, Imaging, Pictures

Fragment identifier considerations: Fragment identifiers are specified in Annex C of ISO/IEC 14496-12:2022

Additional information:

Magic number(s): none

File extension(s): hmg

Intended usage: Common

### P.3 Examples (informative)

Content-Type:

image/hif2; codecs="vvc1.1.L51.CQA"; profiles="mif3"

An image file with low-overhead HEIF container containing one VVC-coded image using Main 10 profile, Main Tier, Level 3.1.

Content-Type:

image/hif2; codecs="hvc1.A1.80.L93.B0"; profiles="mif3"

An image file with low-overhead HEIF container containing one HEVC-coded image using Main profile, Main tier, Level 3.1.

Replace the following text in Annex C:

Fragment identifier considerations: Fragment identifiers are specified in Annex L of ISO/IEC 14496-12, available as a Publicly Available Standard at http://standards.iso.org/ittf/PubliclyAvailableStandards/index.html

with

Fragment identifier considerations: Fragment identifiers are specified in Annex C of ISO/IEC 14496-12:2022.

Add the following entries to Bibliography:

[13] ICC.1:2001-04, *File format for color profiles*, International Color Consortium

Add the following entries to Clause 2 Normative References:

ISO 15076-1, *Image technology colour management — Architecture, profile format and data structure — Part 1: Based on ICC.1:2010*