Text

Description automatically generatedISO/IEC JTC 1/SC 29/WG 03 N1405

**ISO/IEC JTC 1/SC 29/WG 03  
MPEG Systems   
Convenorship: KATS (Korea, Republic of)**

**Document type:** Output Document

**Title:** **Potential Improvement Text of FDAM ISO/IEC 23001-17 DAM 2 Generic compression of samples and items in ISOBMFF**

**Status:** Approved

**Date of document:** 2024-11-05

**Source:** ISO/IEC JTC 1/SC 29/WG 03

**No. of pages:** 25 (with cover page)

**Email of Convenor:** young.L @ samsung . com

**Committee URL:** <https://isotc.iso.org/livelink/livelink/open/jtc1sc29wg3>

**INTERNATIONAL ORGANIZATION FOR STANDARDIZATION**

**ORGANISATION INTERNATIONALE DE NORMALISATION**

**ISO/IEC JTC 1/SC 29/WG 03 MPEG SYSTEMS**

**ISO/IEC JTC 1/SC 29/WG 03 N1405**

**November 2024, Kemer, TR**

|  |  |
| --- | --- |
| **Title** | **Potential Improvement Draft FDAM Text of ISO/IEC 23001-17 DAM 2 Generic compression of samples and items in ISOBMFF** |
| **Source** | **WG 03, MPEG Systems** |
| **Status** | **Approved** |
| **Serial Number** | **24565** |

**ISO/IEC JTC 1/SC 29/WG3 Nnnnn**

Date : 2024-11-08

**ISO/IEC 23001-17:2022/AMD 2:2024(E)**

ISO/IEC JTC1/SC 29

Secretariat: JISC

**Information technology — MPEG Systems technologies — Part 17: Carriage of uncompressed video and images in ISO base media file format — Amendment 2:** **Generic compression of samples and items in ISOBMFF**

FDAM 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.

Introduction

This amendment to ISO/IEC 23001-17 includes the following additions and corrections to the standard.

* Defines the capability to apply generic compression to still and motion imagery as well as other media types (e.g., KLV metadata tracks)
* Clarifies the original standard text related to padding of images using subsampled chroma components
* Corrects a typo in Clause 5.3.2 Table 5
* Adds the capability to encode signed integer components.
* Adds brands for generically compressed media

While 23001-17 nominally defines the mechanism to carry uncompressed imagery and raster data within ISOBMFF, it also more generically defines a mechanism to define the in-memory layout for an item or sample, and then store the image using that layout:

* Component value types and sizes
* Tiling to enable efficient spatial-based access for large imagery
* Padding to ensure that individual component values or groups of component values can be accessed by the processing unit without having to perform bit shifts for every single access, and without having to cross key storage/memory page boundaries

This capability brings ISOBMFF on par with other generic storage formats for numeric data (such as HDF5) with one exception – data-agnostic numerically-lossless and bitwise-lossless compression, transparent to the end user. In HDF5, for example, a large N-dimensional array of numerical data can be chunked (i.e., tiled) and then each chunk is compressed using off-the-shelf ubiquitous data-agnostic compression tools (e.g., deflate). This provides storage and transmission savings similar to numerically lossless image coders, with minimal computational performance impact. These capabilities can be applied not only to typical integer-based pixel formats, but also to IEEE 754 floating-point pixel formats that are unsupported in most imagery compression algorithms.

Adding this capability to 23001-17 is expected to provide cost savings (for both storage and network transmission), particularly for applications and datasets involving large amounts of uncompressed content, such as geospatial and scientific imagery, without significantly changing how those applications access the pixel data.

In this amendment, this mechanism is applied to KLV formatted metadata tracks in addition to image samples and items. Application to other media types can be defined in future standards.

**Use cases**

* Data producer generates a large image (or image sequence) using 32-bit IEEE 754 binary floating point component values after calibration. The image is tiled using 1024 × 1024 tiles and each tile is independently compressed using deflate.
* Data consumer desires to load only a specific spatial region from an image or sample based on some form of chunking of the image (chunk by rectangular tiles or chunk by rows). Consumer uses offset/size information provided within the ISOBMFF file to locate only the desired chunks. Each of those chunks can be independently decompressed.
* Data consumer desires to load only a spatial region from a large tiled image, where the desired region is smaller than a region contained within a single compressed tile. After decompressing the tile, the order, alignment and padding of the component data is maintained, enabling the consumer to calculate individual component value offsets – parsing through the decompressed tile to locate specific pixels/component is not necessary.
* A data producer collects an image of the Earth scanning diagonally from northwest to southeast. For simpler human viewing of the image, the collected image is rotated 45 degrees to align north to up, and additional padding is added to form a tiled, rectangular image buffer. Since the four corner tiles are entirely fill data, they are omitted from the data stored within the ISOBMFF file.
* Legal requirements for records management require bitwise-lossless compression of the image data – it is not sufficient that the decompressed pixel value is numerically equal to the original pixel value; the specific bit patterns must also be the same.

**Requirements**

* Numerically and bitwise lossless compression of items and track samples, especially when consisting of floating-point formatted media.
* Pixel organization prior to compression (as well as post-decompression) is defined by the uncompressed spec in 23001-17. The point is for the encoder to determine how to best organize the pixel/component data when in a directly-accessible form, and then to implement simple off-the-shelf, numerically/bitwise-lossless compression on that data.
* Utilization of existing compression technology with open licensing, broad/mature support, and availability of open-source software and tools.
* Ability to access portions of the image without fully decompressing the entire sample or item.
  + Ability to compress, access, and decompress tiles independently. This includes gridded items as well as tiles defined within 23001-17.
  + Ability to minimize coding/inclusion of fill data.
    - e.g., a large geospatial image is rotated within the rectangular pixel boundaries so north is up. This causes the corners of the expanded rectangular image to be just fill pixels, with a resulting preference to not have to store those fill pixels
    - Specifying a transformative property for arbitrary rotation is ultimately not sufficient as there are many means to precisely georeference an image (e.g., orthorectification) involving pushing pixels around based on imaging geometry and terrain.
    - Determine if sub images of a gridded item can be omitted.
* Orthogonal capability to the 23001-17 component value alignment, padding and component value organization is highly desired. Upon decompression, each chunk is used exactly as if the respective uncompressed values had been loaded directly from storage or the network.
* Constructive interaction between transformational properties is desired. For example, if the sample is compressed as individual chunks, but the compressed sample is then is encrypted as a single chunk, the independence of those chunks might be lost.
* Ability to compress KLV metadata tracks and items.

Contents

[3.12 Generically-compressed 1](#_Toc173153944)

[3.13 Generically-compressed item 1](#_Toc173153945)

[3.14 Generically-compressed sample 1](#_Toc173153946)

[8 Generic compression of items and sample data 4](#_Toc173153947)

[8.1 Overview 4](#_Toc173153948)

[8.2 Compression Configuration Box 5](#_Toc173153950)

[8.2.1 Definition 5](#_Toc173153951)

[8.2.2 Syntax 5](#_Toc173153952)

[8.2.3 Semantics 6](#_Toc173153954)

[8.3 Generically-compressed media tracks 7](#_Toc173153955)

[8.3.1 Overview 7](#_Toc173153956)

[8.3.2 Generically Compressed Units Info Box 7](#_Toc173153958)

[8.4 Generically-compressed items 9](#_Toc173153959)

[8.4.1 Overview 9](#_Toc173153960)

[8.4.2 Generically Compressed Units Item 10](#_Toc173153962)

[8.5 Hypothetical reconstruction model 11](#_Toc173153963)

[9 Mapping to other ISOBMFF capabilities 12](#_Toc173153966)

[9.1 Mapping of generic sample compression to uncompressed image and sample data 12](#_Toc173153967)

[9.2 Combined usage of Generic Sample Compression and Common Encryption 13](#_Toc173153968)

[10 Generic compression brands 13](#_Toc173153969)

[10.1 Brand for generic compression of media tracks 13](#_Toc173153970)

[10.2 Brand for generic compression of items 14](#_Toc173153971)

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 23001 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 — MPEG Systems technologies — Part 17: Carriage of uncompressed video and images in ISO base media file format — Amendment 2: Generic compression for samples and items in ISOBMFF

*Add the following entries into clause 2 (Normative Reference)*

* Deflate compression IETF RFC 1951
* Deflate compression and zlib bitstream format IETF RFC 1950
* Brotli compression IETF RFC 7932

*Add the following new subclauses after subclause 3.11*

## Generically-compressed

compressed using one of a defined set of off-the-shelf numerically and bitwise lossless compression capabilities

## Generically-compressed item

item that has been numerically and bitwise losslessly compressed

## Generically-compressed sample

sample that has been numerically and bitwise losslessly compressed

*In clause 5.2.1.3, Table 2, add the following row and modify the last row to reflect use of the previously reserved value 3:*

**Table 2 - Component formats**

|  |  |
| --- | --- |
| Value | Description |
| 3 | Component value is a signed two’s complement integer coded on component\_bit\_depth bits. For this component format, component\_bit\_depth values shall be greater than 1. |
| other values | ISO/IEC reserved for future definition |

*In clause 5.2.1.5.3,* *replace the following lines:*

If row\_align\_size is not 0 and interleave\_type is 0:

* row\_align\_size shall be a multiple of 2
* the row alignment for components of type ‘U’ and ‘V’, as defined in 5.2.1.7, shall be done using row\_align\_size/2

If tile\_align\_size is not 0:

* tile\_align\_size shall be a multiple of 2
* the tile alignment for components of type ‘U’ and ‘V’, as defined in 5.2.1.7, shall be done using tile\_align\_size/2

*with the following (the second bullet in each list is deleted):*

If row\_align\_size is not 0 and interleave\_type is 0:

* row\_align\_size shall be a multiple of 2

If tile\_align\_size is not 0:

* tile\_align\_size shall be a multiple of 2

*In clause 5.2.1.5.4,* *replace the following lines:*

If row\_align\_size is not 0 and interleave\_type is 0:

* row\_align\_size shall be a multiple of 2
* the row alignment for components of type ‘U’ and ‘V’, as defined in 5.2.1.7, shall be done using row\_align\_size/2

If tile\_align\_size is not 0:

* tile\_align\_size shall be a multiple of 4
* the tile alignment for components of type ‘U’ and ‘V’, as defined in 5.2.1.7 shall be done using tile\_align\_size/4

*with the following (the second bullet in each list is deleted):*

If row\_align\_size is not 0 and interleave\_type is 0:

* row\_align\_size shall be a multiple of 2

If tile\_align\_size is not 0:

* tile\_align\_size shall be a multiple of 4

*In clause 5.2.1.5.5, replace the following lines:*

If row\_align\_size is not 0 and interleave\_type is 0:

* row\_align\_size shall be a multiple of 4
* the row alignment for components of type ‘U’ and ‘V’, as defined in 5.2.1.7, shall be done using row\_align\_size/4

If tile\_align\_size is not 0:

* tile\_align\_size shall be a multiple of 4
* the tile alignment for components of type ‘U’ and ‘V’, as defined in 5.2.1.7, shall be done using tile\_align\_size/4

*with the following (the second bullet in each list is deleted):*

If row\_align\_size is not 0 and interleave\_type is 0:

* row\_align\_size shall be a multiple of 4

If tile\_align\_size is not 0:

* tile\_align\_size shall be a multiple of 4

*In clause 5.2.1.7, 2nd paragraph after NOTE4, replace:*

Rows of tiles shall be byte-aligned at the end of the row:

*with the following:*

Rows and tile rows shall be byte-aligned at the end of the row:

*In clause 5.2.1.7, replace:*

If row\_align\_size is 0, no additional padding is present at the end of rows of tiles. Otherwise, let RowSize be the number of bytes required to contain, for a given row R:

* all values of all components of row R if interleave\_type is 1 or 5 or if interleave\_type is 2 and component type is ‘U’ or ‘V’ (including all component, block and pixel padding within and at the end of the sample data for row R),
* all values of the current component for row R (including all component and block padding within the sample data for row R) otherwise.

*with the following:*

If row\_align\_size is 0, no additional padding is present at the end of rows and tile rows. Otherwise, let RowSize be the number of bytes required to contain, for a given row R:

* all values of all components of row R (including all component, block and pixel padding within and at the end of the sample data for row R) if interleave\_type is 1 or 5
* all values of pixel interleaved components (‘U’ and ‘V’) of row R if interleave\_type is 2 and component type is ‘U’ or ‘V’ (including all component, block and pixel padding within and at the end of the sample data for row R),
* all values of the current component for row R (including all component and block padding within the sample data for row R) otherwise.

*Correct the last row of Table 5 in Clause 5.3.2 to be the following. Specifically, the string within the “Field values” column should be ‘yv20’ to match the profile identifier.*

**Table 5 – Predefined uncompressed frame formats**

|  |  |  |
| --- | --- | --- |
| Profile identifier | Description | Field values for UncompressedFrameConfigBox |
| … | | |
| 'yv20' | YUV 420 8 bits planar YCrCb | {'yv20', [{1,7},{3,7},{2,7}], 2, 0} |

*Add following new clauses after clause 7*

# Generic compression of items and sample data

## Overview

Storing uncompressed item and sample data is required in many use cases, some of which are not well-supported by typical image compression algorithms (e.g., floating point or very high-bit depth imagery). Other types of sample data, such as KLV (SMPTE 336M), are only compressible by standard generic data compression mechanisms. The ability to compress data for any media type without compression loss is desirable for reducing storage sizes and transmission times, such as in the following scenarios:

* A data producer generates a large image (or image sequence) using 32-bit IEEE 754 binary floating point component values after calibration. The image is tiled using 1024 × 1024 tiles and each tile is independently compressed using deflate.
* A data consumer desires to load only a specific spatial region from an image or sample based on some form of chunking of the image (chunk by rectangular tiles or chunk by rows). Consumer uses offset/size information provided within the file to locate only the desired chunk(s). Each chunk is independently decompressed.
* A data consumer desires to load only a spatial region from a large tiled image, where the desired region is smaller than a region contained within a single compressed chunk. After decompressing the chunk, the order, alignment and padding of the component data is maintained, enabling the consumer to locate individual component values via calculated offsets.
* A data producer collects an image of the Earth scanning diagonally from northwest to southeast. For simpler human viewing of the image, the collected image is rotated 45 degrees to align north to up, and additional padding is added to form a tiled, rectangular image buffer. Since the four corner tiles are entirely fill data, they can more efficiently be stored than simply compressing multiple blocks of fill data.
* A data producer generates multiple blocks of metadata, encoded using Key-Length-Value encoding, and stores them as individual items. The value portion of each KLV metadata item is a set or pack containing binary metadata and is generically compressed to save space.

To decompress a complete media sample or item, the file reader locates the data as given by the sample or item’s compressed units description. The extracted compressed data is then decompressed according to the compression algorithm specified by the compression\_type field in the CompressionConfigurationBox. The resultant data is formatted exactly as was specified by the underlying media format, including any padding placed at the end of the element to align the next element.

If the value of the compressed\_unit\_type field in the CompressionConfigurationBox is not 0, decompression of individual portions of the sample is possible. In those cases, the individual ranges specified in the GenericallyCompressedUnitsInfoBox map to the individual units as specified by the compressed\_unit\_type field, in the order those units would have been found in the coded data were that coded data left uncompressed as specified in Clause 5.

For media tracks, generic compression is signalled using a restricted transformation scheme as specified in Clause 8.3, and can be used in conjunction with other track transformation such as encryption or other restricted video transformation, as specified in ISOBMFF.

For items, generic compression is signalled using an essential property as specified in Clause 8.4 and can be used in conjunction with item protection as defined in ISOBMFF.

## Compression Configuration Box

### Definition

Box Type: 'cmpC'  
Container: SchemeInformationBox or ItemPropertyContainerBox   
Mandatory: Yes (when the SchemeType is 'gcmp')  
Quantity: One

The CompressionConfigurationBox specifies the specific data compression method used and codec-specific type of compressed units within a media sample or item data.

This box can be:

* added to a restricted video sample entry for media tracks
* added as properties associated with an item.

The syntax in this section is given for a video sample entry container and the defined box therefore extends FullBox. When used in an ItemPropertyContainerBox, the same syntax applies but the defined boxes extends ItemFullProperty. The properties defined in the following sections are descriptive properties.

The definition of each value of compression\_type specifies not only the algorithm but also the bitstream format for each compressed subsample. For example, ‘zlib’ specifies the use of the deflate algorithm as packaged in the zlib format defined by IETF RFC 1950.

Value 0 for compressed\_unit\_type indicates that the compressed range is always the complete media sample or item.

Derived specifications may assign compressed\_unit\_type values other than 0 according to the specificities of the underlying media format.

### Syntax

aligned(8) class CompressionConfigurationBox extends FullBox('cmpC', version=0, flags=0) {  
 unsigned int(32) compression\_type;  
 unsigned int(8) compressed\_unit\_type;  
}

### Semantics

compression\_type is a 4CC indicating the compression mode for the sample or item. Values are defined in Table 6.

**Table 6 – Compression types**

| Value | Description |
| --- | --- |
| ‘defl’ | DEFLATE algorithm as defined in IETF RFC 1951 |
| ‘zlib’ | DEFLATE algorithm as packaged in the format defined by IETF RFC 1950 |
| ‘brot’ | Brotli algorithm as defined in IETF RFC 7932 |
| ‘iden’ | Identity (no compression) |

compressed\_unit\_type indicates the unit being compressed within a generically-compressed media sample or item. The range of bytes specified for each unit contains exactly the result of the compression algorithm applied to the identified unit.

A value of 0 means that compression is applied to the entire sample or item. Other values of the compressed\_unit\_type are specified in Clause 9.2. For example, if the value of this field is 2 (units are tiles), then each tile of the item is independently accessible and decompressible.

NOTE: Depending on the compression algorithm and codestream format, a reader might be able to decompress multiple units as a single combined input buffer. For example, while the IETF RFC 1950 format for DEFLATE does include a short header at the start of each compressed unit, the decompressor does recognize that header even if it is in the middle of the compressed data buffer. In many cases, it might be possible for units to be concatenated in any order and input to a single instance of the decompressor as long as the reader knows what order the units were found in the concatenated compressed buffer, and can then extract those units from the decompressed buffer. However, this capability is not guaranteed by this standard.

## Generically-compressed media tracks

### Overview

Generically-compressed media tracks compliant to this specification are media tracks compliant to ISO/IEC 14496-12 that use a reserved transformation scheme of type 'gcmp', hereafter called generically-compressed sample entry. The scheme version for the 'gcmp' scheme shall be set to 1.

The SchemeInformationBox of a generically-compressed sample entry shall contain one CompressionConfigurationBox, specifying the data compression method used and the partitioning of the compression scheme over the sample data.

The GenericallyCompressedUnitsInfoBox is used to specify the location and size of each independently compressed element within the sample.

For non-fragmented movies, the sample table of a track using a generically-compressed sample entry shall contain one GenericallyCompressedUnitsInfoBox. For a fragmented movie, each TrackFragmentBox of a track using a generically-compressed sample entry shall contain one GenericallyCompressedUnitsInfoBox.

A generically-compressed sample entry does not impact the description of the media samples of the track: random access information, subsample description and sample group properties describe the media sample without the generic compression being applied.

If content encryption is applied, it shall be applied on the compressed sample. If the protection scheme uses a track transformation (e.g. Common Encryption), the sample entry shall describe these two transformations as specified in ISOBMFF.

### Generically Compressed Units Info Box

#### Definition

Box Type: 'cbri'  
Container: SampleTableBox or TrackFragmentBox  
Mandatory: See below  
Quantity: Zero or One

The GenericallyCompressedUnitsInfoBox describes the units of compressed data for samples of a track or track fragment.

The GenericallyCompressedUnitsInfoBox shall be present whenever the track has a sample entry using a restricted scheme of type 'gcmp'and the compressed\_unit\_type of the associated CompressionConfigurationBox has a value other than 0.

The sample size, as documented by the SampleSizeBox or TrackRunBox, shall be the size of the compressed material described by the GenericallyCompressedUnitsInfoBox, i.e. shall be equal to MAX(unit\_offset + unit\_size)of any compressed unit in the sample. If no compressed unit is present for a sample, the entire sample payload is the compressed unit, i.e. an implicit unit is defined with unit\_size equal to the sample size and unit\_offset equal to 0.

Each compression unit of a sample shall be compliant to the compression type indicated by the CompressionConfigurationBox of the sample entry this sample is mapped to.

All samples in the track or track fragment shall be documented in the Generically­Compressed­Units­Info­Box. Within a sample, it is possible to mutualize identically compressed data by using the same entity offset and size as a previously described entity.

NOTE 1: A compressed unit of a sample can reuse parts of the sample data by pointing to the same data using the same unit\_offset.

NOTE 2: For a given sample, the compressed unit size/offset pairs are listed in the GenericallyCompressedUnitsInfoBox in the order specified for that unit type (as specified by the compression\_unit\_type field in the CompressionConfigurationBox). For example, when units correspond to tiles, the unit (i.e., tile) size/offset pairs are listed in raster-scan order as specified in clause 5.2.1.4. However, the actual data for those units within the sample data can be in any order. Derived specifications may further constrain this, for example to require that the compressed data for each compressed unit be in the same order as the units.

#### Syntax

aligned class compressed\_unit\_info(unit\_offset\_nbbits, unit\_size\_nbbits) {  
  
 if (unit\_offset\_nbbits)  
 unsigned int(unit\_offset\_nbbits) unit\_offset;

unsigned int(unit\_size\_nbbits) unit\_size;  
}

class GenericallyCompressedUnitsInfoBox extends FullBox('cbri', version=0, flags=0) {

unsigned int(3) unit\_offset\_code;

unsigned int(3) unit\_size\_code;

bit(2) reserved = 0;

unsigned int(32) num\_samples;

int unit\_offset\_nbbits;

if (unit\_offset\_code==0) unit\_offset\_nbbits=0;

else if (unit\_offset\_code==1) unit\_offset\_nbbits=16;

else if (unit\_offset\_code==2) unit\_offset\_nbbits=24;

else if (unit\_offset\_code==3) unit\_offset\_nbbits=32;

else if (unit\_offset\_code==4) unit\_offset\_nbbits=64;

int unit\_size\_nbbits;

if (unit\_size\_code==0) unit\_size\_nbbits=8;

else if (unit\_size\_code==1) unit\_size\_nbbits=16;

else if (unit\_size\_code==2) unit\_size\_nbbits=24;

else if (unit\_size\_code==3) unit\_size\_nbbits=32;

else if (unit\_size\_code==4) unit\_size\_nbbits=64;

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

unsigned int(32) num\_compressed\_units;

for (int r = 0; r < num\_compressed\_units; r++) {

compressed\_unit\_info(

unit\_offset\_nbbits,

unit\_size\_nbbits

);

}

}

}

#### Semantics

unit\_offset\_code indicates the number of bits used to describe unit offsets. A value of 0 implies that offsets are not used.

unit\_size\_code indicates the number of bits used to describe unit sizes.

num\_samples is an integer indicating the number of samples for the current entry. This value shall not be 0.

num\_compressed\_units is an integer indicating the number of compressed units for samples described by the current entry. If this value is 0, this implies that a single unit covering the entire sample data is used.

NOTE: If the structure of the sample varies within the track or track segment (e.g., the dimensions or size of the sample changes), then the number of compressed units might also change depending on how elements within the sample are mapped to units.

compressed\_unit\_info.unit\_offset is the offset in bytes to the first byte of the compressed unit from the start of the sample data as stored in the file. If unit\_offset\_nbits is 0, this field is not coded and takes the value:

* 0 for the first compressed unit of the sample
* Sum of all compressed\_unit\_info.unit\_size values of all the preceding compressed units of the sample otherwise

compressed\_unit\_info.unit\_size is size in bytes of this compressed unit. This value shall not be 0

## Generically-compressed items

### Overview

Generically-compressed items have an associated essential item property of type 'cmpC', which shall be marked as essential. When content protection is applied to a generically-compressed item, the protection shall be applied to the compressed payload. Generically-compressed items can be used as can be any other items without restriction, including as part of an image grid item.

### Generically Compressed Units Item

#### Definition

Box Type: 'icef'  
Container: ItemPropertyContainerBox   
Mandatory: No  
Quantity: Zero or One

The GenericallyCompressedUnitsItemInfoBox describes the units of compressed data for an item.

The GenericallyCompressedUnitsItemInfoBox shall be present for any item with an associated essential property of type 'cmpC' (CompressionConfigurationBox) for which compressed\_unit\_type has a value other than 0. When not present, this implies that a single unit covering the complete item data is used.

The item size in bytes is the size of the compressed material described by the Generically­Compressed­Units­ItemInfo­Box, i.e. shall be equal to MAX(unit\_offset + unit\_size)of any compressed unit in the item. This size shall be equal to the sum of all extent\_length of the extents describing this item in ItemLocationBox.

Each compressed unit of an item shall be compliant to the compression type indicated by the CompressionConfigurationBox associated to this item. There is no requirement that each compressed unit corresponds to an extent of the item in the ItemLocationBox.

NOTE 1: A compressed unit of an item can reuse parts of the item data by pointing to the same data using the same unit\_offset. Similarly, an item may reuse parts of another item using the extent construction mechanisms.

NOTE 2: For a given item, the compressed unit size/offset pairs are listed in the GenericallyCompressedUnitsItemInfoBox in the order specified for that unit type (as specified by the compression\_unit\_type field in the CompressionConfigurationBox). For example, when units correspond to tiles, the unit (i.e., tile) size/offset pairs are listed in raster-scan order as specified in clause 5.2.1.4. However, the actual data for those units within the item data can be in any order. Derived specifications may further constrain this, for example to require that the compressed data for each compressed unit be in the same order as the units.

If this property is not associated with an item using generic compression, the entire item is the compressed unit.

#### Syntax

The dependent class compressed\_unit\_info is as defined in clause 8.3.2.

aligned(8) class GenericallyCompressedUnitsItemInfoBox extends  
ItemFullProperty('icef', version=0, flags=0) {

unsigned int(3) unit\_offset\_code;

unsigned int(3) unit\_size\_code;

bit(2) reserved = 0;

int unit\_offset\_nbbits;

if (unit\_offset\_code==0) unit\_offset\_nbbits=0;

else if (unit\_offset\_code==1) unit\_offset\_nbbits=16;

else if (unit\_offset\_code==2) unit\_offset\_nbbits=24;

else if (unit\_offset\_code==3) unit\_offset\_nbbits=32;

else if (unit\_offset\_code==4) unit\_offset\_nbbits=64;

int unit\_size\_nbbits;

if (unit\_size\_code==0) unit\_size\_nbbits=8;

else if (unit\_size\_code==1) unit\_size\_nbbits=16;

else if (unit\_size\_code==2) unit\_size\_nbbits=24;

else if (unit\_size\_code==3) unit\_size\_nbbits=32;

else if (unit\_size\_code==4) unit\_size\_nbbits=64;

unsigned int(32) num\_compressed\_units;

for (int r = 0; r < num\_compressed\_units; r++) {

compressed\_unit\_info(unit\_offset\_nbbits, unit\_size\_nbbits);

}

}

#### Semantics

unit\_offset\_code indicates the number of bits used to describe unit offsets. A value of 0 implies that offsets are not used.

unit\_size\_code indicates the number of bits used to describe unit sizes.

num\_compressed\_units is an integer that indicates the number of compressed units for the item. This value shall not be 0.

compressed\_unit\_info.unit\_offset is the offset in bytes to the first byte of the compressed unit, with value 0 being the first byte of the item payload. If unit\_offset\_nbits is 0, this field is not coded and takes the value:

* 0 for the first compressed unit of the item
* Sum of all unit\_size values of the preceding compressed units otherwise.

The bytes for a compressed unit shall be contained in a single item extent. One item extent may however contain more than one compressed unit.

compressed\_unit\_info.unit\_size is size in bytes of this compressed unit. This value shall not be 0.

## Hypothetical reconstruction model

The reconstruction by a file reader of an entire generically compressed item or sample shall be equivalent to the result of the following process:

* Identify if generic compression is used:
  + Presence of CompressionConfigurationBox associated property for an item
  + Presence of a ‘resX’ sample entry transformation with a scheme type ‘gcmp’, implying a CompressionConfigurationBox in the SchemeInformationBox
* Identify the compression type and unit type in the CompressionConfigurationBox
* Extract the item or sample payload as for any uncompressed item or payload, using extents for items and sample size and offsets for media tracks
* If Common Encryption applies to the sample, decrypt the entire sample payload as defined in Common Encryption
* Locate the GenericallyCompressedUnitsItemInfoBox for items, or the entry for the given sample in the GenericallyCompressedUnitsInfoBox for media tracks
* Prepare an empty decompression buffer for the item/sample
* For each compressed unit, in the order they are listed
  + Extract the bytes from the item or sample payload using the unit\_offset and unit\_size of the compressed unit
  + Decompress these bytes using the identified decompressor
  + Concatenate these bytes into the decompression buffer
* Replace the item/sample payload with the decompression buffer and update the item/sample size as seen by the application

Implementations can of course find better ways to optimize this process, especially if accessing only a subset of the compressed units is desired (e.g. individually compressed tiles), as indicated by the CompressionConfigurationBox. In such cases, an implementation can decide to (decrypt and) decompress only the bytes required for its needs.

# Mapping to other ISOBMFF capabilities

## Mapping of generic sample compression to uncompressed image and sample data

For uncompressed video sample entries or items compliant to this document, the following compressed\_unit\_type values for the CompressionConfigurationBox are defined in Table 7.

**Table 7 – Compressed unit types**

|  |  |
| --- | --- |
| Value | Description |
| Types applicable to all items and samples | |
| 0 | the unit shall be the full item or sample as defined in clause 8.2.3 |
| Image-related types | |
| 1 | the unit shall be the full image for a given component (component-based interleave and mixed interleave) |
| 2 | the unit shall be a single tile |
| 3 | the unit shall be a single row |
| 4 | the unit shall be a single pixel |
| other values | ISO/IEC reserved for future definition |
| Other types | |
| other values | ISO/IEC reserved for future definition |

Unit ranges are specified in the same order as the units themselves. For example, the unit ranges for image tiles are in the same order as the tiles as specified in Clause 5.

NOTE: Compressed unit types are potentially applicable to other types of data besides items and samples and thus specified independently from the values for the compressed\_unit\_type field in the CompressionConfigurationBox in clause 8.2.

## Combined usage of Generic Sample Compression and Common Encryption

When a media sample or item is encrypted using ISO/IEC 23001-7 Common Encryption, the encryption boundaries, i.e. the start of a CENC subsample, should be aligned with significant semantic elements of the item, e.g. start of a tile, a row, a component plane, etc. Given the uncompressed nature of the media, it is expected that most CENC subsamples will have a BytesOfClearData with value 0. Derived specifications may further restrict the mapping of these elements to CENC subsamples.

When a media sample or item is using both generic sample compression and encryption using ISO/IEC 23001-7 Common Encryption, the following apply:

* The encryption shall be applied to the generic compressed data,
* The first byte of each compressed unit shall be the first byte of a CENC subsample which shall have no decryption dependency to previous CENC subsamples. This implies that a full crypt block starts on this first byte, and that either CTR block chaining is used or that the cypher state is re-initialized at the start of the subsample (i.e. ‘cbcs’ scheme).

A media track using both generic compression and common encryption will therefore have two sample entry transformations, first for encryption then for generic compression. For a video track, this can be summarized as:

* Sample entry type ‘encv’
* ProtectionSchemeInfoBox
  + - SchemeType ‘cenc’, ‘cbcs’ …
    - SchemeInfoBox→TrackEncryptionBox (CENC specific )
    - OriginalFormatBox indicating ‘gcmp’
  + RestrictedSchemeInfoBox
    - SchemeType ‘gcmp’
    - SchemeInfoBox→CompressionConfigurationBox
    - OriginalFormatBox indicating original media format (‘uncv’ for example)

# Generic compression brands

## Brand for generic compression of media tracks

The brand 'gcmm' shall be used to indicate compatibility with generically compressed media samples and items. The 'gcmm' brand requires support for the 'iso6' brand.

Processing of restricted sample entries (i.e. 'resv') with a scheme type of 'gcmp' is required under this brand.

Additionally, the 'gcmm' brand requires support for:

* CompressionConfigurationBox with version = 0 and flags = 0
* GenericallyCompressedUnitsInfoBox with version = 0 and flags = 0
* The compression types 'defl' and 'iden'

## Brand for generic compression of items

The brand 'gcmi' shall be used to indicate compatibility with generically compressed media samples and items. The 'gcmi' brand requires support for the 'isoa' brand.

Additionally, the 'gcmi' brand requires support for:

* CompressionConfigurationBox with version = 0 and flags = 0
* GenericallyCompressedUnitsItemInfoBox with version = 0 and flags = 0
* The compression types 'defl' and 'iden'