Text

Description automatically generatedISO/IEC JTC 1/SC 29/WG 03 N0747

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

**Document type:** Output Document

**Title:** Preliminary WD of 14496-12 8th Edition AMD 1 Support for T.35, original sample duration and other improvements

**Status:** Approved

**Date of document:** 2022-11-30

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

**No. of pages:** 10 (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 N0747**

**November 2022, Mainz**

|  |  |
| --- | --- |
| **Title** | **Preliminary WD of 14496-12 8th Edition AMD 1 Support for T.35, original sample duration and other improvements** |
| **Source** | **WG 03, MPEG Systems** |
| **Status** | **Approved** |
| **Serial Number** | **22074** |

# Introduction

This amendment addresses the following topics:

* T.35 signaling in ISOBMFF [m61234](https://dms.mpeg.expert/doc_end_user/current_document.php?id=84729): [GitLab](http://mpegx.int-evry.fr/software/MPEG/Systems/FileFormat/isobmff/-/issues/154)
* Original Duration signaling [m61144](https://dms.mpeg.expert/doc_end_user/current_document.php?id=84639): [GitLab](http://mpegx.int-evry.fr/software/MPEG/Systems/FileFormat/isobmff/-/issues/149)
* Efficient subsample [m61170](https://dms.mpeg.expert/doc_end_user/current_document.php?id=84665): [GitLab](http://mpegx.int-evry.fr/software/MPEG/Systems/FileFormat/isobmff/-/issues/151)

# Changes to 14496-12

***Add a new Clause 10.13 (T.35 sample group)***

## 10.13 T.35 sample group

### 10.13.1 Definition

Each sample of a track may be associated with zero or more sample group descriptions, each of which defines a record of T.35 information of different types. The same T.35 information may apply to different samples.

The grouping\_type=‘it35’ is defined as a grouping criterion for T.35 metadata. The SampleTableBox or TrackFregmentBox of a track can contain zero or more SampleToGroupBoxes or CompactSampleToGroupBoxes with the grouping\_type='it35'. Each sample group description shall document exactly one 'kind' of T.35 message, i.e. shall map to T.35 sample group entries with at least the same ITU-T T.35 *country code*, *terminal provider code* and *terminal provider oriented code,* and T.35 type and version (if applicable). The placement of T.35 metadata within the samples is out of scope and should be specified in derived specifications.

NOTE 1 both, *terminal provider code* and *terminal provider oriented code* are stored within first bytes of the itu\_t\_t35\_header. Also, the signaling of the T.35 version may appear after *terminal\_provider\_oriented\_code*. The number of bytes used to identify the origin and version of the T.35 metadata depends on the country and the entity which defines the syntax of the T.35 message.

Sample group descriptions with grouping\_type='it35' shall use version 1 or above.

[Ed Note: It needs to be clarified how the versions of the sample group description box and the sample to group box are used in different scenarios. Also, the use of the grouping\_type\_parameter should be clarified if version 1 is allowed.]

### 10.13.2 Syntax

class T35SampleGroupEntry extends SampleGroupDescriptionEntry('it35') {  
 bit(8) itu\_t\_t35\_country\_code;  
 if(itu\_t\_t35\_country\_code == 0xFF)  
 {  
 bit(8) itu\_t\_t35\_country\_code\_extension\_byte;  
 }  
 bit(8) itu\_t\_t35\_data[];  
}

### 10.13.3 Semantics

itu\_t\_t35\_country\_code shall be a byte having a value specified as a country code by Rec. ITU-T T.35 Annex A, or the country code extension value 0xFF.

itu\_t\_t35\_country\_code\_extension\_byte if present, shall be a byte having a value specified as a country code by Rec. ITU-T T.35 Annex B.

itu\_t\_t35\_data shall be the payload containing data registered as specified in Rec. ITU-T T.35.

The ITU-T T.35 *terminal provider code* and *terminal provider oriented code* shall be contained in the first one or more bytes of the itu\_t\_t35\_header, in the format specified by the Administration that issued the terminal provider code. It should also include a part of the ITU-T T.35 message which specifies the version of the message and may include any remaining data of the ITU-T T.35 message having syntax and semantics as specified by the entity identified by the ITU-T T.35 *country code*, *terminal provider code* and *terminal provider oriented code*.

The length of this field is the number of bytes remaining in the SampleGroupDescriptionEntry, with the size specified in the parent SampleGroupDescriptionBox.

***Add a new Clause 8.11.17 (T.35 item)***

**8.11.17 T.35 item**

**8.11.17.1 Definition**

A T.35 metadata item carries an ITU-T T.35 message. When the T.35 metadata is stored as metadata item the item\_type value shall be 'it35'.

**8.11.17.2 Syntax**

aligned(8) class T35Information {  
 bit(8) itu\_t\_t35\_country\_code;  
 if(itu\_t\_t35\_country\_code == 0xFF)  
 {  
 bit(8) itu\_t\_t35\_country\_code\_extension\_byte;  
 }  
 bit(8) itu\_t\_t35\_payload[];  
}

**8.11.17.3 Semantics**

itu\_t\_t35\_country\_code shall be a byte having a value specified as a country code by Rec. ITU-T T.35 Annex A, or the country code extension value 0xFF.

itu\_t\_t35\_country\_code\_extension\_byte if present, shall be a byte having a value specified as a country code by Rec. ITU-T T.35 Annex B.

itu\_t\_t35\_payload shall be the payload containing data registered as specified in Rec. ITU-T T.35.

The ITU-T T.35 *terminal provider code* and *terminal provider oriented code* shall be contained in the first one or more bytes of the itu\_t\_t35\_payload, in the format specified by the Administration that issued the *terminal provider code*. Any remaining itu\_t\_t35\_payload data shall be data having syntax and semantics as specified by the entity identified by the ITU-T T.35 *country code*, *terminal provider code* and *terminal provider oriented code*.

The length of this field is the number of bytes remaining in the item.

***Add a new Clause 8.8.18 (Original timing signaling)***

## 8.8.18 Redundant SampleOriginal Timing

**8.8.18.1 Definition**

Box Type: 'rsot'  
Container: TrackFragmentBox  
Mandatory: No  
Quantity: Zero or one

The RedundantSampleOriginalTimingBox can be used to document that the first sample of the track fragment is a copy of the previous sample, if any, and that this sample’s original start time was intended to be before its actual sample decode time. This allows rewinding the sample playback time at tune-in but ignoring it in regular playback mode.

The RedundantSampleOriginalTimingBox can also be used to document that the duration of the last sample of a fragment was truncated to respect fragmentation constraints, and that the intended duration of the sample is longer than its actual duration in the fragment; this allows exact processing of samples for which internal timing logic is dependent on the sample duration (such as animations).

The following flags are defined:

* RSOT\_ORIGINAL\_DURATION: flag value is 0x000001. If set, the original duration of the last sample in the track fragment is documented
* RSOT\_ELAPSED\_DURATION: flag value is 0x000002. If set, the elapsed duration of the first sample in the track fragment is documented

When flag RSOT\_ORIGINAL\_DURATION is set, it indicates that the last sample of the track fragment has a shorter duration than originally authored, and the original duration is signaled. The originalDuration shall be equal to or greater than the duration of the last sample in the track fragment.

When flag RSOT\_ELAPSED\_DURATION is set, the first sample of the track fragment is treated as if its associated sample\_flags value has sample\_depends\_on=2 and sample\_has\_redundancy=1, and the flag documents the first sample of the track fragment is a copy of the previously received sample if any. If no previous sample was received for this track (tune in), the first sample is processed at its sample decode time as if it was being presented for the indicated elapsedDuration. Otherwise (a previous sample was already received for this track), the duration of the previous sample is extended by the duration of this first sample and the elapsedDuration is ignored. If the previous sample had an originalDuration *OrigDur* signaled, the extended duration shall be:

* Strictly less than *OrigDur* if this is the only sample of the track fragment with a FragmentedSampleOriginalTimingBox present with flag RSOT\_ORIGINAL\_DURATION set,
* Equal to *OrigDur* otherwise

NOTE RSOT\_ORIGINAL\_DURATION and RSOT\_ELAPSED\_DURATION may be set together in a track fragment with multiple samples (describing that first sample is a continuation and last sample is truncated) or with a single sample (describing the only sample is both a continuation and truncated, for example when splitting a sample of 20 second long into three or more fragments).

**8.8.18.2 Syntax**

aligned(8) class RedundantSampleOriginalTiming extends FullBox('rsot', 0, flags)  
{  
 if (flags & RSOT\_ORIGINAL\_DURATION)  
 unsigned int(32) originalDuration;  
 if (flags & RSOT\_ELAPSED\_DURATION)  
 unsigned int(32) elapsedDuration;  
}

**8.8.18.3 Semantics**

originalDuration gives the original duration of the last sample of the track fragment, in media timescale of this sample. Value shall be different from 0.

elapsedDuration gives the elapsed duration of the first sample of the track fragment, in media timescale of this sample. Value shall be different from 0.

***Add a new Clause 8.7.10 (Efficient subsample)***

## 8.7.10 Sub-sample Reference Table Box

**8.7.10.1 Definition**

Box Type: 'ssrt'  
Container: SampleTableBox or TrackFragmentBox  
Mandatory: No  
Quantity: Zero or one

This box is designed to contain a list of properties of sub-samples that are likely to occur several times.

**8.7.10.2 Syntax**

aligned(8) class SubSampleReferenceTableBox   
 extends FullBox('ssrt', version = 0, flags) {  
 unsigned int(1) is\_short\_ref;  
 if (is\_short\_ref) {  
 reference\_length = 7;  
 } else {  
 reference\_length = 15;  
 }  
 unsigned int(reference\_length) entry\_count;  
 for (int i = 0; i < entry\_count; i++) {  
 unsigned int(1) discardable;  
 unsigned int(1) has\_subsample\_priority;  
 unsigned int(1) is\_csp\_present;  
 unsigned int(2) csp\_length\_minus\_1;  
 unsigned int(3) reserved;  
 if (has\_subsample\_priority) {  
 unsigned int(8) subsample\_priority;  
 }  
 if (is\_csp\_present) {  
 unsigned int(8\*( csp\_length\_minus\_1 + 1))  
 codec\_specific\_parameters;  
 }  
 }  
}

**8.7.10.3 Semantics**

is\_short\_ref equal 1 indicates that there are at most 128 entries comprised in SubSampleReferenceTableBox, in which case 7 bits are used to encode the index of an entry in said table. The value 0 indicates at most 32.768 entries are comprised in the reference table, in which case 15 bits are used to encode the index of an entry in said table.

entry\_count is an integer that gives the number of entries in SubSampleReferenceTableBox.

discardable equal to 0 indicates that the sub-sample is required to decode the current sample, while equal to 1 indicates the sub-sample is not required to decode the current sample but may be used for enhancements, e.g., the sub-sample consists of supplemental enhancement information (SEI) messages.

has\_subsample\_priority equal to 1 indicates that subsample\_priority value is present. The value equal to 0 indicates that no subsample\_priority value is present.

is\_csp\_present equal to 1 indicates that codec\_specific\_parameters is present. The value 0 indicates no codec\_specific\_parameters value is present.

csp\_length\_minus\_1 indicates the number of bytes, minus 1, used to encode the codec\_specific\_parameters value.

subsample\_priority is an integer specifying the degradation priority for each sub-sample. Higher values of subsample\_priority, indicate sub-samples which are important to, and have a greater impact on, the decoded quality. When not present, its value is inferred to 0.

codec\_specific\_parameters is defined by the codec in use. If no such definition is available, this field shall be set to 0. When not present, its value is inferred to 0.

***Add to Clause 8.7.7.1 (Efficient subsample)***

Sub-sample information box provides 2 ways to describe sub-sample information:

* Version 0 and 1 explicitly describe all the properties of each sub-sample;
* Version 2 enables describing some of the properties for some sub-samples through a reference to a sub-sample description stored in a SubSampleReferenceTableBox.

[Ed. Note: The versioning aspect needs further investigation. The current definition of the subs box covers future (undefined) versions which could lead to possible problems in implementations.]

***Replace Clause 8.7.7.2 with (Efficient subsample)***

aligned(8) class SubSampleInformationBox   
extends FullBox('subs', version, flags) {  
 unsigned int(32) entry\_count;  
 if (version == 2) {  
 unsigned int(1) is\_size\_16\_bits;  
 unsigned int(1) is\_short\_ref;  
 if (is\_short\_ref) {  
 reference\_length = 7;  
 } else {  
 reference\_length = 15;  
 }  
 unsigned int(6) reserved;  
 }  
 for (int i = 0; i < entry\_count; i++) {  
 unsigned int(32) sample\_delta;  
 unsigned int(16) subsample\_count;  
 if (subsample\_count > 0) {  
 for (int j = 0; j < subsample\_count; j++) {  
 if (version == 2) {  
 if (is\_size\_16\_bits) {  
  unsigned int(16) subsample\_size;  
 } else {  
 unsigned int(32) subsample\_size;  
 }  
 unsigned int(1) has\_reference;  
 if (has\_reference) {  
 unsigned int(reference\_length) reference\_idx;  
 } else {  
 unsigned int(1) discardable;  
 unsigned int(1) has\_subsample\_priority;  
 unsigned int(1) is\_csp\_present;  
 unsigned int(2) csp\_length\_minus\_1;  
 unsigned int(2) reserved;  
 if (has\_subsample\_priority) {  
 unsigned int(8) subsample\_priority;  
 }  
 if (is\_csp\_present) {  
 unsigned int(8\*(csp\_length\_minus\_1+1))  
 codec\_specific\_parameters;  
 }  
 }  
 else {  
 if (version == 1) {  
 unsigned int(32) subsample\_size;  
 } else {  
 unsigned int(16) subsample\_size;  
 }  
 unsigned int(8) subsample\_priority;  
 unsigned int(8) discardable;  
 unsigned int(32) codec\_specific\_parameters;  
 }  
 }  
 }  
 }  
}

***Add yellow highlighted parts to Clause 8.7.7.3 (Efficient subsample)***

version is an integer that specifies the version of this box (0, 1 or 2 in this document).

entry\_count is an integer that gives the number of entries in the following table.

is\_size\_16\_bits indicates, in version 2 of this box, whether subsample\_size is encoded on 16 or 32 bits. When equal to 1, subsample\_size is coded on 16 bits. When equal to 0, subsample\_size is coded on 32 bits.

is\_short\_ref equals 1 indicates that there are at most 128 entries in SubSampleReferenceTableBox, in which case 7 bits are used to encode the index of an entry in said table. The value 0 indicates at most 32.768 entries are comprised in the reference table, in which case 15 bits are used to encode the index of an entry in said table.

sample\_delta is an integer that indicates the sample having sub‐sample structure. It is coded as the difference, in decoding order, between the desired sample number, and the sample number indicated in the previous entry. If the current entry is the first entry in the track, the value indicates the sample number of the first sample having sub-sample information, that is, the value is the difference between the sample number and zero (0). If the current entry is the first entry in a track fragment with preceding non-empty track fragments, the value indicates the difference between the sample number of the first sample having sub-sample information and the sample number of the last sample in the previous track fragment. If the current entry is the first entry in a track fragment without any preceding track fragments, the value indicates the sample number of the first sample having sub-sample information, that is, the value is the difference between the sample number and zero (0). This implies that the sample\_delta for the first entry describing the first sample in the track or in the track fragment is always 1.

subsample\_count is an integer that specifies the number of sub-sample for the current sample. If there is no sub-sample structure, then this field takes the value 0.

subsample\_size is an integer that specifies the size, in bytes, of the current sub-sample.

has\_reference equal to 1 indicates that the values of these properties are identical to the properties in the reference\_idx-th entry of SubSampleReferenceTableBox. When equal to 0, it indicates that subsample\_priority, discardable and codec\_specific\_parameters are explicitly indicated for the sub-sample. In this case, the number of bits reference\_length that is used to indicate a reference is determined based on the value of is\_short\_ref.

reference\_idx is an integer that specifies the index of an entry in the associated SubSampleReferenceTableBox. reference\_idx is a 0-based index whose value shall be lower than the number of entries in SubSampleReferenceTableBox.

has\_subsample\_priority equal to 1 indicates that subsample\_priority value is present. The value equal to 0 indicates that no subsample\_priority value is present.

is\_csp\_present equal to 1 indicates that codec\_specific\_parameters is present. The value 0 indicates no codec\_specific\_parameters value is present.

csp\_length\_minus\_1 indicates the number of bytes, minus 1, used to encode the codec\_specific\_parameters value.

subsample\_priority is an integer specifying the degradation priority for each sub-sample. Higher values of subsample\_priority, indicate sub-samples which are important to, and have a greater impact on, the decoded quality. When not present, its value is inferred to 0.

discardable equal to 0 means that the sub-sample is required to decode the current sample, while equal to 1 means the sub-sample is not required to decode the current sample but may be used for enhancements, e.g., the sub-sample consists of supplemental enhancement information (SEI) messages.

codec\_specific\_parameters is defined by the codec in use. If no such definition is available, this field shall be set to 0. When not present, its value is inferred to 0.