SMPTE 384M-2005

SMPTE STANDARD

for Television — Material Exchange Format (MXF) — Mapping of Uncompressed Pictures into the Generic Container

1 Scope

This standard specifies the mapping of uncompressed pictures into the MXF generic container. This MXF body is suitable for the mapping of all sampling grids and sampling formats including but not limited to those defined by ITU-R BT.601, ITU-R BT 656, ITU-R BT 709, SMPTE 125M, SMPTE 267M, SMPTE 274M, and SMPTE 296M. Provision is made for mapping of the active picture area alone or of larger sampling grids, including some or all of the horizontal and vertical ancillary data areas. Provision is made for alignment of the stored sampling grid to memory block size boundaries. This standard defines SMPTE universal labels that uniquely identify specific uncompressed implementations.

This standard defines uncompressed picture essence elements that may be used as elements in content packages within an MXF generic container.

In order to achieve interoperability within any given operational pattern, restrictions may be placed on the way in which this essence container can be implemented. The reader is advised to carefully study the appropriate operational pattern document before implementation.

2 Normative references

SMPTE 291M-1998, Television — Ancillary Data Packet and Space Formatting

SMPTE 336M-2001, Television — Data Encoding Protocol Using Key-Length-Value

SMPTE 352M-2002, Television (Dynamic) — Video Payload Identification for Digital Television Interfaces

SMPTE 377M-2004, Television — Material Exchange Format (MXF) — File Format Specification

SMPTE 379M-2004, Television — Material Exchange Format (MXF) —MXF Generic Container

SMPTE RP 210, Metadata Dictionary Registry of Metadata Element Descriptions

SMPTE RP 224, Registry of SMPTE Universal Labels

3 Glossary of acronyms, terms and data types

The full glossary of acronyms terms and data types used in the MXF specification is given in the MXF format specification. A supplementary glossary of acronyms and terms is defined in SMPTE 379M. They are not repeated here to avoid any divergence of meaning. Terms defined in this specification:

DID	Data Identification (see SMPTE 291M)
SDID	Secondary Data Identification (see SMPTE 291M)

Approved
September 2, 2005

4 Introduction

This standard is concerned primarily with the wrapping of uncompressed pictures so that they may be carried in MXF. This primarily involves the arrangement of the physical bytes within the essence container and the values for KLV keys and labels. The parameters that may be used to describe the picture information, in terms of the picture dimensions, sampling, pixel data representation and underlying signal standard are fully described in the MXF format document annexes D and E. The principles of wrapping and underlying concepts of the generic container as well as the outline key values are defined in the generic container specification. Indeed the principle of signaling with the essence container UL is outlined in these two documents and is used in this standard to signal that a well known uncompressed picture format is being transported within the essence container. To fully understand this document, both the MXF format specification and generic container specification need to be understood.

This standard uses the generic term "uncompressed picture" to refer to any variant of picture sampling (horizontal, vertical and temporal), sampling structure, and picture color space, including but not limited to those defined by ITU-R BT.601, ITU-R BT 656, ITU-R BT 709, SMPTE 125M, SMPTE 267M, SMPTE 274M, and SMPTE 296M.

This standard uses the term "ancillary data area" to refer to any sampled data within the captured signal that is outside the active image area defined for the uncompressed picture. This standard and all MXF documents use the phrase “sampling grid” is used to mean a captured spatial arrangement of pixels, usually specified in a standard.

This standard defines uncompressed picture essence elements that may be used as elements in content packages within an MXF generic container.

Provision is made for mapping of the active picture area alone, or of larger sampling grids including some or all of the horizontal and vertical ancillary data areas. Provision is made for padding of the stored data to memory block size boundaries in order to accommodate picture formats that are already in use.

When uncompressed pictures are used within an MXF generic container, the essence shall be divided into elements of equal duration, corresponding to one sample unit. The duration of the sample unit is determined by the sample rate property of the essence descriptor. The relationship between sample units and the stored image rectangle is defined by the frame layout property of the essence descriptor and covers progressive frame, interlaced fields, and segmented frame. It is possible that an MXF file may contain a single picture or a sequence of two or more pictures.

5 Mapping uncompressed pictures to a file structure (informative)

5.1 Basic mapping of uncompressed pictures

Each uncompressed picture is mapped into a content package of the essence container where each content package has the duration of one edit unit. The uncompressed picture can be made up of fill data, ancillary information, picture information and other data. The terminology used is defined in an annex of the MXF format specification.

The essence container may comprise a single content package (where the single uncompressed picture is a still) or a continuous sequence of two or more content packages. In the latter case, the format of each uncompressed picture should be consistent if continuous decoding of the essence is required. Note that fill data and ancillary data may vary from content package to content package while maintaining continuous decoding.

Figure 1 is reproduced from the MXF file format specification in order to help the reader of this standard with some of the terminology. The definition of the terms in the diagram is extensively discussed in the MXF format specification.

Note specifically that when the stored, sampled and displayed rectangles are different, the stored data may include Ancillary data along with the displayed image pixels. The line wrapping mode allows this ancillary data to have its own KLV key.

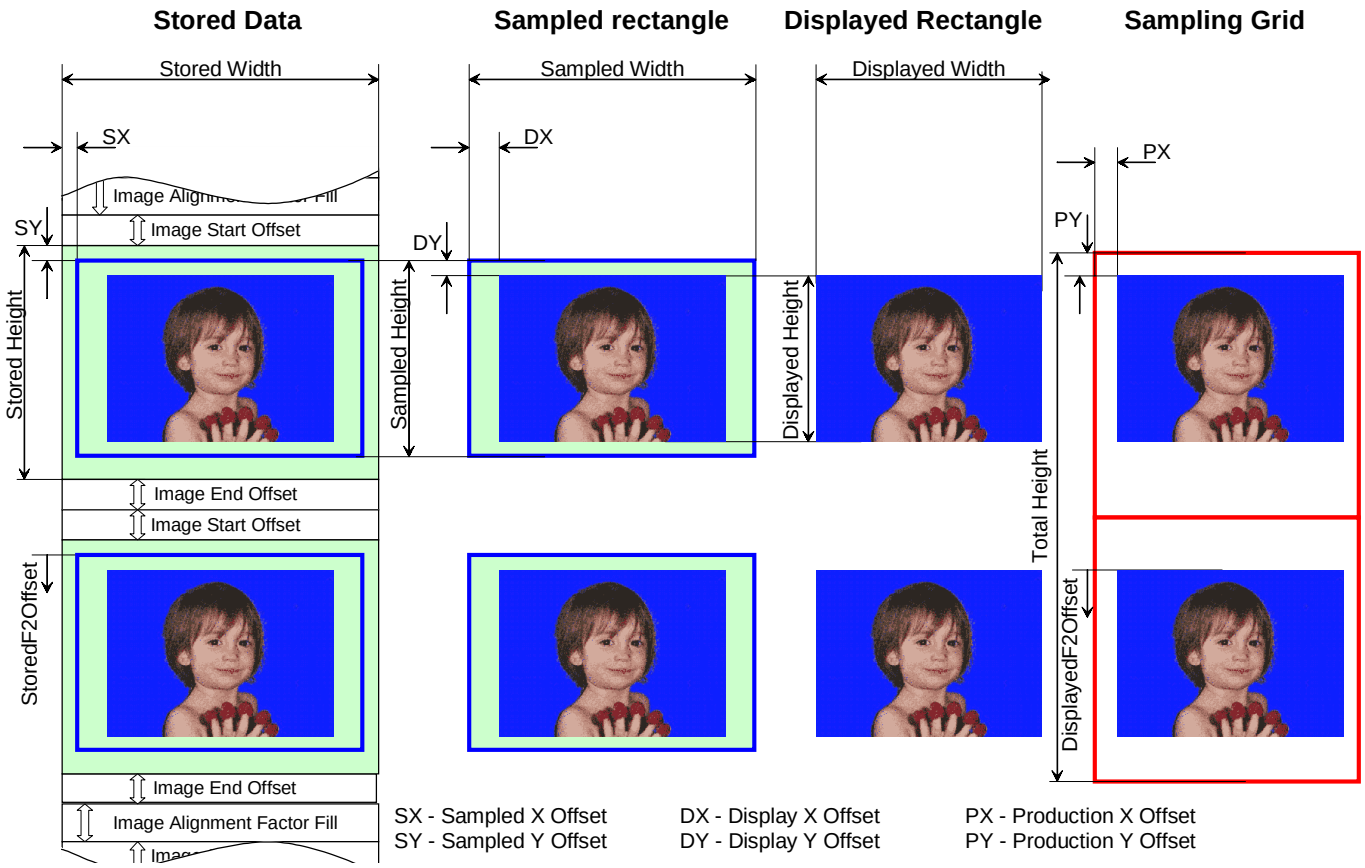

Figure 1 – Stored, sampled, displayed rectangles and sampling grid
(Informative figure copied from format document)

6 Mapping the uncompressed picture data to the MXF generic container

Three types of KLV wrapping are specified for the uncompressed picture body structure.

- Frame-based wrapping
- Clip-based wrapping
- Line-based wrapping

A sequence of pictures shall be KLV coded as defined in SMPTE 336M. The ancillary and picture data bytes for a generic single frame (or field) are shown pictorially in figure 2. For clarity, the sampled rectangle is not considered. VANC is intended to show a vertical data area (i.e., the offset from the 1st stored line to the 1st displayed line). HANC is intended to show a horizontal data area (i.e., the offset from the left-most stored pixel to the left-most displayed pixel). Stored bytes before and after the picture are represented as $Offset_s$ and $Offset_e$ respectively.

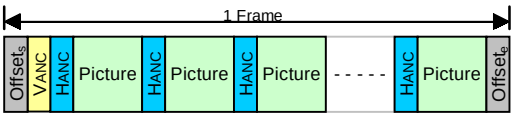

Figure 2 – Simple representation of an uncompressed frame

6.1 Frame-based wrapping

SMPTE 379M defines frame-based mapping. The "frame wrapping" method for uncompressed pictures is shown in figure 3. This figure shows each picture wrapped in a content package with no other elements in the container.

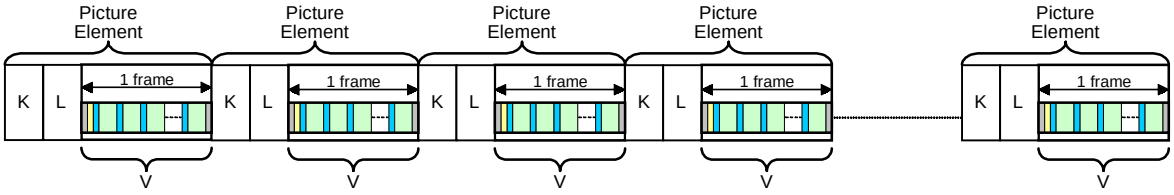

Figure 3 – Simple representation of frame wrapping

The frame wrapping method enables frame-by-frame access by MXF applications that process at the KLV level. This can be particularly useful for applications that support multiple generic container mapping types. Sufficient Information is provided to allow individual frames to be identified at the KLV level without an MXF decoder having to parse or decode the essence data. Each frame of uncompressed picture data is KLV wrapped using a GC picture element key.

In some applications, the frame wrapped uncompressed picture information will exist in content packages with other elements such as sound and data elements. An example generic container is shown in figure 4.

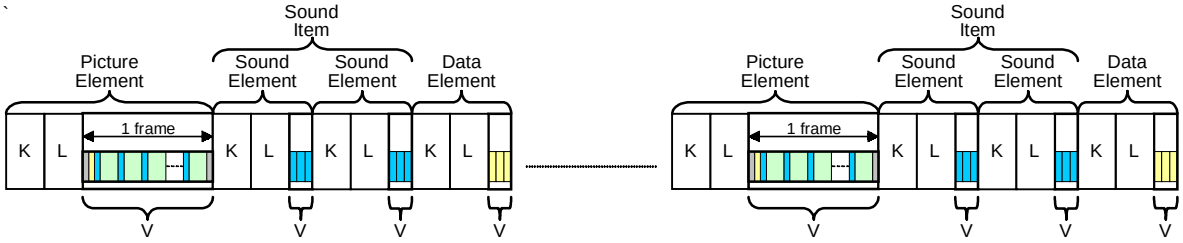

Figure 4 – Frame wrapping with other GC elements

The generic container mapping specifications for the sound and data elements define the key values and format of the data within the elements. Note that in this wrapping mode, the sound and data elements should have a duration of one frame. This is different to the line wrapping mode below that is intended to KLV wrap the individual components of the uncompressed picture data in the order in which they are stored/transmitted.

6.2 Clip-based wrapping

SMPTE 379M defines clip-based mapping. The clip wrapping method for uncompressed pictures is shown in figure 5. KLV encoding wraps the whole of the MXF uncompressed picture body that may contain one or more frames (maybe thousands). Any other elements in this generic container should also be clip wrapped.

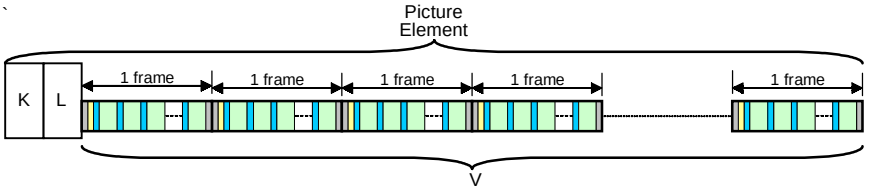

Figure 5 – Simple representation of clip wrapping

The clip wrapping method is for applications that carry the uncompressed picture data as a single large entity. This can be very useful applications such as store and forward servers that process whole files and also in applications where it is desired to use the rich metadata structures of MXF as an annotation to uncompressed picture data. The clip of uncompressed picture data is KLV wrapped using a picture element key. When uncompressed picture data is clip wrapped, there shall be only one clip per generic container body. Multiple clips can be concatenated and edited using the operational pattern mechanism detailed in the MXF format document.

In some applications, the clip wrapped uncompressed picture data will exist in a content package with other elements such as sound and data elements. An example generic container is shown in figure 6.

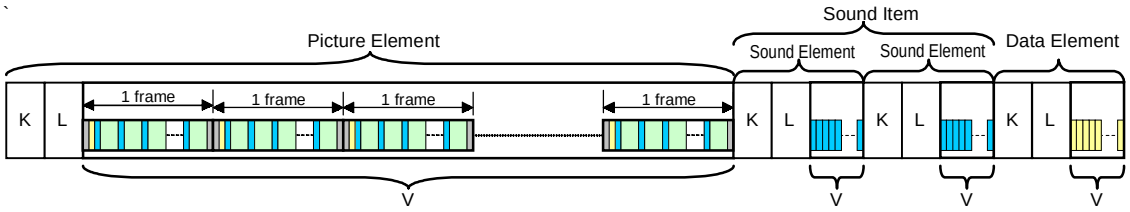

Figure 6 – Clip wrapping with other GC elements

The generic container mapping specifications for the sound and data elements shall detail the key values and format of the data within the elements. Note that in this wrapping mode, the sound and data elements should have a duration of the entire clip.

6.3 Line-based wrapping

Line-based mapping is defined in this standard. In line-based wrapping, KLV encoding wraps each individual component of each line of the MXF uncompressed picture. This identifies fills and ancillary data separately from lines within the stored data. Care must be taken when using this mode to ensure that the resulting file is generic container compliant. The generic container rules constraining the grouping of elements into items shall be met.

The uncompressed picture from figure 2 is shown line wrapped in figure 7. The generic container recommends that the data within each content package should represent a similar time duration. In this wrapping mode, each content package shall correspond to a single line of picture data. Note that extra data

before the first line is associated with the first line's content package. Extra data after the last line shall be associated with the last line of the picture.

The generic container requires that all elements of any item in a content package shall be uniquely identified. It is also strongly recommended that elements within an item are adjacent within any content package. There are occasions when this might not be possible, for example the HANC data element and $Offset_e$ data element in the final content package in figure 7. When the individual elements are wrapped, care must be taken to comply with this recommendation.

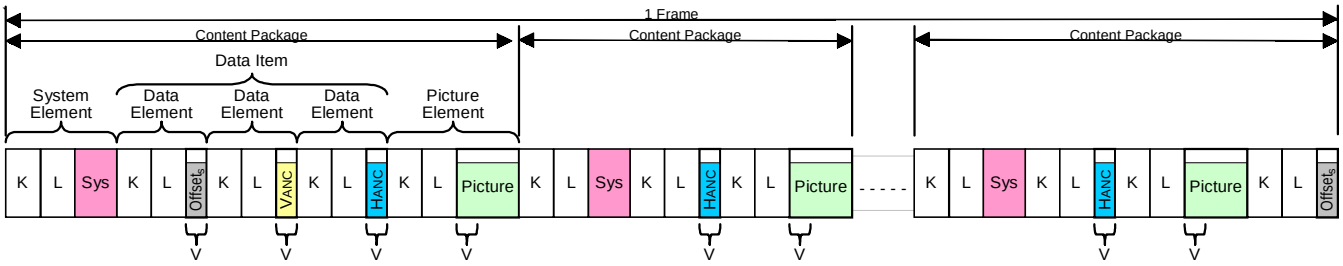

Figure 7 – Line wrapping of an uncompressed picture

If the type of ancillary data is known then the key shall be correctly set. If the ancillary data does not conform to a known standard, or cannot be identified when the file is created, then the line wrapped data element type value in table 2 shall be used.

Line-based wrapping is very likely to result in a generic container with content packages comprising different numbers of elements. This dynamic structure may make it difficult to uniquely identify the start of a content package. For this reason, a simple system item is defined that identifies the start of each line.

7 Keys for GC elements when wrapping uncompressed pictures

As can be seen from section 6, there is always a picture element in the three wrapping modes. In line-wrapping mode, there may additionally be extra GC elements. The keys for these elements are defined in table 2 in section 7.2.

7.1 Picture element key

The values of the last four bytes of the essence element key are defined in table 1.

Table 1 – Key value for the uncompressed picture element

Byte No.	Description	Value (hex)	Meaning
1-12	See MXF Generic Container Specification		
13	Item Type Identifier	15h	Picture Item
14	Essence Element Count	kkh	Count of Picture Elements in this Item
15	Essence Element Type	02h 03h 04h	Frame Wrapped Picture Element Clip Wrapped Picture Element Line Wrapped Picture Element (as listed in SMPTE RP 224)
16	Essence Element Number	nnh	The Number (used as an Index) of this Picture Element in this Item

7.1.1 Picture element length

The length field of the KLV coded element shall be 4 bytes BER long-form encoded (i.e., 83h.xx.yy.zz) for frame or line-based wrapping. The length field of the KLV coded element shall be 8 bytes BER long-form encoded (i.e., 87h.aa.bb.cc.dd.ee.ff.gg) for clip wrapping.

7.1.2 Essence element count — Byte 14

This is a count of the number of elements in the picture item of the content package.

7.1.3 Essence element type — Byte 15

For frame wrapped uncompressed pictures, the value shall be 02h. For clip wrapped uncompressed pictures, the value shall be 03h. For line wrapped uncompressed pictures, the value shall be 04h.

7.1.4 Essence element number — Byte 16

This is a number used as an index to identify this instance of the element type within the item in each content package. The generic container specification requires that this byte be unique within each item of a content package. Each and every instance of the uncompressed picture element shall retain a consistent essence element number throughout the entire generic container.

7.2 Line-wrapped data element keys

These keys shall only be used for the line wrapping mode. There are many different types of data that may be mapped into the HANC and VANC regions of an uncompressed picture. Generic keys are provided for broad categorization of this data. The values of the last four bytes of the essence element key are defined in table 2.

Table 2 – Key value for the line-wrapped uncompressed picture data element

Byte No.	Description	Value (hex)	Meaning
1-12	See MXF Generic Container Specification		
13	Item Type Identifier	17h	GC Data Item
14	Essence Element Count	kkh	Count of Data Elements in this Item
15	Essence Element Type	08h 09h 0Ah	Line Wrapped Data Element Line Wrapped VANC Data Element Line Wrapped HANC Data Element (as listed in SMPTE RP 224)
16	Essence Element Number	nnh	The Number (used as an Index) of this Data Element in this Item

When line wrapping is used and SMPTE 291M compliant ancillary data is found, the MXF encoder shall insert a new data element key for every ancillary data flag found. For each combination of DID and SDID found in the stream, there shall be a single, unique data element key value.

Note that there may be some pathological cases where the arrangement of video, audio and data in the uncompressed stored data prevents this rule from being followed. In such extremely rare cases, generic container compliant line wrapped files cannot be constructed. A different wrapping mechanism shall be employed to be compliant with the specification.

7.2.1 Data element length

The length field of the KLV coded element shall be 4 bytes BER long-form encoded (i.e., 83h.xx.yy.zz).

7.2.2 Essence element count — Byte 14

This is a count of the number of elements in the data item of the content package.

7.2.3 Essence element type — Byte 15

For general line wrapped data (e.g., any $Offset_s$ or $Offset_e$ data), the value shall be 08h. For general line wrapped VANC data, the value shall be 09h. For general line wrapped HANC data, the value shall be 0Ah.

7.2.4 Essence element number — Byte 16

This is a number used as an index to identify this instance of the element type within the item in each generic container. The generic container specification requires that this byte be unique within each item of a generic container. This mapping document recommends that each and every different component of the line shall retain a consistent essence element number throughout the entire generic container. This will help decoder implementations recognize data components that occur on many lines or in many frames. For example, any $Offset_s$ data should have the same essence element number in each and every occurrence of the element within the entire generic container.

7.3 System element

This key shall only be used for the line-based wrapping mode. It is intended to uniquely identify the start of a line. The values of the last four bytes of the system element key are defined in table 3.

Table 3 – Key value for the line-wrapped uncompressed picture system element

Byte No.	Description	Value (hex)	Meaning
1-12	See MXF Generic Container Specification (note that byte 6 should be set to the value indicating fixed length pack)		
13	Item Type Identifier	14h	GC System Item
14	System Scheme Identifier	01h	Uncompressed picture Line wrapped system scheme
15	System Element Identifier	01h 02h	Start of Stored Data Start of Line See SMPTE 379M
16	Reserved for use by metadata Element	00h	reserved

7.3.1 System scheme identifier — Byte 14

This identifies the system element and shall have the value 01h.

7.3.2 Metadata or control element identifier — Byte 15

This identifies the start of the stored data with the value 01h or the start of a line (left hand edge of the stored data) with the value 02h.

7.3.3 System element length

The length field of the KLV coded element shall be 4 bytes BER long-form encoded (i.e., 83h.xx.yy.zz).

7.3.4 System element value

The line wrapped uncompressed picture system element shall be a fixed length pack with the following structure defined in table 4.

Table 4 – Uncompressed picture system element structure

	Item Name	Type	Len	UL Designator	Meaning	Default
	System Element	Set Key	16	Table 3	Identifies the System Element	
↔	Length	BER Length	4		Overall Length of System Element Value	8
	Line number	UInt32	4	04.01.03.02. 0A.00.00.00	The line number of this line of the Picture	
	SMPTE 352M payload ID	Structure	4	04.01.05.04. 02.00.00.00	The SMPTE 352M payload ID	

Note that if byte 15 of the key has the value 01h then the line number shall be 01h.

The SMPTE 352M payload ID shall be present in each and every system element, even if it is also present as an ancillary data packet within the essence. If there is no 352M payload ID for the image format being mapped, then the value 0 shall be used in the SMPTE 352M payload ID.

7.4 Other elements in the generic container

7.4.1 Fill elements

There may be fill information present in the generic container. The KLV fill metadata item is defined in the SMPTE metadata dictionary (SMPTE RP 210).

NOTE – The universal label of the KLV fill metadata item has the value:

06h.0Eh.2Bh.34h. 01h.01h.01h.01h. 03h.01h.02h.10h. 01h.00h.00h.00h

7.4.2 Other elements

Other elements may be present in frame wrapped or clip wrapped MXF generic containers. These elements, such as sound and data elements, are defined in separate essence mapping documents. There shall be no extra elements in a line wrapped MXF generic container.

7.4.3 Use of generic container system elements

When it is desired to use MXF files containing uncompressed pictures in streaming environments, the MXF generic container should include a system element that carries the SMPTE 352M signal type identifier where it exists.

7.4.4 Use of KAG

There are no specific KAG requirements for this mapping. MXF encoders and decoders shall comply with the KAG rules in the MXF format document.

8 SMPTE label for essence container identification

The values for the container UL are given in table 5.

Table 5 – Specification of the uncompressed picture essence container label

Byte No.	Description	Value (hex)	Meaning
1-12	Defined by Generic Container		
13	Essence Container Kind	02h	MXF Generic Container
14	Mapping Kind	05h	Uncompressed Pictures as listed in SMPTE RP 224
15	Locally defined	xxh	Number of lines / field rate combination see Table 6
16	Locally defined	yyh	See Table 7

This SMPTE label is the individual 'essence container' property used in the partition pack, in the preface set and in the appropriate file descriptor.

Byte 14 identifies the container as mapping uncompressed pictures into the generic container.

Byte 15 is intended to enumerate the number of lines and field rate combinations within the picture. common values are listed in SMPTE RP 224. When a non-standard combination is used, this byte shall have the value 7Fh as defined in table 6 that indicates the parameters shall be obtained from the appropriate essence descriptor.

Table 6 – Specification of the uncompressed picture container label, byte 15

Byte 15	Meaning
0h	not used
xxh	see SMPTE RP 224
7Fh	Number of Lines and Frame rate must be determined from the Essence Descriptor

Byte 16 is intended to carry basic information about the containment of the uncompressed picture. It is set according to table 7.

Table 7 – Specification of the uncompressed picture container label, byte 16

Byte 16	Meaning
0h	not used
01h	Frame Wrapping
02h	Clip Wrapping
03h	Line Wrapping
04h-7Fh	Reserved

9 Essence descriptors for uncompressed pictures

Picture essence descriptors for uncompressed pictures shall be carried in the header metadata for the MXF file as described for each operational pattern.

Picture essence descriptors shall be one of the two subclasses of picture essence descriptor that are defined:

- **CDCIDescriptor** – (Color-Difference Component Image Descriptor) for 4:2:2 (YC_bC_r) sampling and similar formats.

- **RGBADescriptor** – (Red Green Blue Alpha Descriptor) for computer graphics sampling and for 4:4:4:4 sampling, and similar formats.

These descriptors are defined in the MXF format specification, SMPTE 377M. The status of each of the items within the descriptor shall be the same as the MXF format specification.

Annex A (informative)
Index tables for uncompressed pictures

In all of these index tables, it is assumed the reader has a good knowledge of the generic index table specification in the MXF format specification. The constraints in this annex shall be the implementation of those tables for uncompressed pictures.

A.1 Frame wrapping with no other elements in the generic container

This is a simple case where the index table points to the first byte of the uncompressed picture element generic container key. There are no other generic container items. The only pieces of information in the index table segment are the start position, duration and (fixed) size of each uncompressed picture element KLV triplet.

Table A.1 – Frame wrapped index table segment set

Item Name	Req ?	Meaning	Use
Index Table Segment	Req	An Index Table Segment set	See MXF Format Specification
Length	Req	Set Length	See MXF Format Specification
Instance ID	Req	Unique ID of this instance	See MXF Format Specification
Index Edit Rate	Req	Edit Rate copied from the tracks of the Essence Container	See MXF Format Specification
Index Start Position	Req	The first editable unit indexed by this Index Table segment measured in File Package Edit Units	
Index Duration	Req	Time duration of this table segment measured in Edit Units of the referenced Package	
Edit Unit Byte Count	D/Req	Defines the byte count of each and every Edit Unit. A value of 0 indicates that the byte count of Edit Units is defined in the Index Entry Array	Set to the number of bytes in every KLV, including the length of the Key and Length. The Index Table can be used to find the first byte of the KLV of every uncompressed frame
IndexSID	D/Req	Stream Identifier (SID) of Index Table	See MXF Format Specification
BodySID	Req	Stream Identifier (SID) of the indexed Essence Container	See MXF Format Specification

A.2 Clip wrapping with no other elements in the generic container

This is an even simpler case where the index table points to the first byte of the payload of the uncompressed picture element KLV triplet. There are no other generic container items within the container. There is only one KLV item.

Table A.2 – Clip wrapped index table segment set

Item Name	Req ?	Meaning	Use
Index Table Segment	Req	An Index Table Segment set	See MXF Format Specification
Length	Req	Set Length	See MXF Format Specification
Instance ID	Req	Unique ID of this instance	See MXF Format Specification
Index Edit Rate	Req	Edit Rate copied from the tracks of the Essence Container	See MXF Format Specification
Index Start Position	Req	The first editable unit indexed by this Index Table segment measured in File Package Edit Units	
Index Duration	Req	Time duration of this table segment measured in Edit Units of the referenced Package	
Edit Unit Byte Count	D/Req	Defines the byte count of each and every Edit Unit. A value of 0 indicates that the byte count of Edit Units is defined in the Index Entry Array	Set to the number of bytes in the uncompressed frame. The Index Table can be used to find the first byte of the stored data for every uncompressed frame
IndexSID	D/Req	Stream Identifier (SID) of Index Table	See MXF Format Specification
BodySID	Req	Stream Identifier (SID) of the indexed Essence Container	See MXF Format Specification

A.3 Frame wrapping with extra items in the generic container

This is a more complex case where the index table points to the first byte of the uncompressed picture element generic container key. The other generic container elements should be indexed by correct use of the delta entries and index entries.

Table A.3 – Frame wrapped index table segment set example for figure 4

	Item Name	Req ?	Meaning	Use
	Index Table Segment	Req	An Index Table Segment set	See MXF Format Specification
↔	Length	Req	Set Length	See MXF Format Specification
	Instance ID	Req	Unique ID of this instance	See MXF Format Specification
	Index Edit Rate	Req	Edit Rate copied from the tracks of the Essence Container	See MXF Format Specification
	Index Start Position	Req	The first editable unit indexed by this Index Table segment measured in File Package Edit Units	
	Index Duration	Req	Time duration of this table segment measured in Edit Units of the referenced Package	
	Edit Unit Byte Count	D/Req	Defines the byte count of each and every Edit Unit. A value of 0 indicates that the byte count of Edit Units is defined in the Index Entry Array	0 unless the total length of all the GC Elements are of constant size. In this example for Figure 4 we assume that only the Data Elements are VBR so the value is 0.
	IndexSID	D/Req	Stream Identifier (SID) of Index Table	See MXF Format Specification
	BodySID	Req	Stream Identifier (SID) of the indexed Essence Container	See MXF Format Specification
	Slice Count	D/Req	Number of slices minus 1 (NSL)	0
	Delta Entry Array	Opt	Map Elements onto Slices	Table A.4
	Index Entry Array	D/Req	Index from Edit Unit number to stream offset	Table A.5

The delta entry array shall contain an entry for every indexed element in the generic container. The order of the elements in the delta entry array shall match the order of the elements in the generic container. The example below is a delta entry array designed to match the example in figure 4. Implementations should construct a delta entry array according to the properties of the actual essence in the file.

In frame-wrapping mode, the element delta values shall include the lengths of the "KL" for each element. The result is that each element delta shall point to the first byte of the key in the KLV that wraps an element. If the overall length of all the elements in each frame is constant, then a delta entry array and an "edit unit byte count" item are sufficient to define the index table segment. In this example, we will assume that the data element in figure 4 is of variable length so that an index entry array is required.

Table A. 4 – Frame wrapped delta entry array example for figure 4

	Field Name	Type	Meaning	Use
	NDE	UInt32	Number of delta entries	4
	Length	UInt32	Length of each delta entry	6
Picture Delta Entry	PosTableIndex	Int8	0= No reordering +ve = PosTable Index	0
	Slice	UInt8	Slice number in IndexEntry	0
	Element Delta	UInt32	Delta from start of slice to this Element	0
Sound Delta Entry	PosTableIndex	Int8	0= No reordering +ve = PosTable Index	0
	Slice	UInt8	Slice number in IndexEntry	0
	Element Delta	UInt32	Delta from start of slice to this Element	sizeof(KL) + sizeof(Picture)
Sound Delta Entry	PosTableIndex	Int8	0= No reordering +ve = PosTable Index	0
	Slice	UInt8	Slice number in IndexEntry	0
	Element Delta	UInt32	Delta from start of slice to this Element	sizeof(KL) + sizeof(Picture) + sizeof(Sound Element 1)
Data Delta Entry	PosTableIndex	Int8	0= No reordering +ve = PosTable Index	0
	Slice	UInt8	Slice number in IndexEntry	0
	Element Delta	UInt32	Delta from start of slice to this Element	sizeof(KL) + sizeof(Picture) + sizeof(Sound Element 1) + sizeof(Sound Element 2)

Table A.5 – Frame wrapped index entry array example for figure 4

	N	Field Name	Type	Meaning	Use
	1	NIE	UInt32	Number of index entries	=number of frames
	1	Length	UInt32	Length of each index array entry	
One Index Entry for every frame	NIE	Temporal Offset	Int8	Offset in edit units from Display Order to Coded Order	0
		Anchor Offset	Int8	Offset in edit units to previous Anchor Frame. The value is zero if this is an anchor frame.	0
		Flags	EditUnitFlag	Flags for this Edit Unit Bit 7: Random Access Bit 6: Sequence Header Bit 5: forward prediction flag Bit 4: backward prediction flag e.g. 00== I frame (no prediction) 10== P frame(forward prediction from previous frame) 01== B frame (backward prediction from future frame) 11== B frame (forward & backward prediction) Bits 0-3: reserved [RP210 Flags to indicate coding of elements in this edit unit]	80h
		Stream Offset	UInt64	Offset in bytes from the first KLV element in this Edit Unit within the Essence Container Stream	Offset from the first byte of the key of the KLV for the first frame to the first byte of the Key of the KLV for the Picture Item in this frame as shown in Figure 4
		SliceOffset	NSL x UInt32	The offset in bytes from the Stream Offset to the start of this slice.	Optional depending on the complexity of the VBR items. In this case it would not be present because there is only 1 slice

A.4 Clip wrapping with extra items in the generic container

This is the most complex case where the index table points to the first byte of the payload of each element in the generic container key. The other generic container elements should be indexed by correct use of the delta entries and index entries. The delta entry values are limited to a UINT32 number range and will therefore be incapable of specifying the offset from the picture item of frame 1 to the sound item of frame 1.

To overcome this, the index table segment should be sliced, even if the offset between elements is constant. This gives, in effect, a 33-bit variable offset from the uncompressed picture item to the corresponding sound item (i.e., a 32-bit variable slice offset + 32-bit fixed delta entry of, for example, 2^{32}-1). The clip-wrapped generic container must be split into several smaller generic containers, each with a single clip.

Table A.6 – Clip wrapped index table segment set example for figure 6

Item Name	Req ?	Meaning	Use
Index Table Segment	Req	An Index Table Segment set	See MXF Format Specification
Length	Req	Set Length	See MXF Format Specification
Instance ID	Req	Unique ID of this instance	See MXF Format Specification
Index Edit Rate	Req	Edit Rate copied from the tracks of the Essence Container	See MXF Format Specification
Index Start Position	Req	The first editable unit indexed by this Index Table segment measured in File Package Edit Units	
Index Duration	Req	Time duration of this table segment measured in Edit Units of the referenced Package	
Edit Unit Byte Count	D/Req	Defines the byte count of each and every Edit Unit. A value of 0 indicates that the byte count of Edit Units is defined in the Index Entry Array	0 unless the total length of all the GC Elements is of constant size. In this example we assume the Data Elements are VBR so the value is 0.
IndexSID	D/Req	Stream Identifier (SID) of Index Table	See MXF Format Specification
BodySID	Req	Stream Identifier (SID) of the indexed Essence Container	See MXF Format Specification
Slice Count	D/Req	Number of slices minus 1 (NSL)	3 (one slice per element)
Delta Entry Array	Opt	Map Elements onto Slices	Table A.7
Index Entry Array	D/Req	Index from Edit Unit number to stream offset	Table A.8

The delta entry array shall contain an entry for every indexed element in the generic container. The order of the elements in the delta entry array shall match the order of the elements in the generic container. The example below is a delta entry array designed to match the example in figure 6. Implementations should construct a delta entry array according to the properties of the actual essence in the file.

In clip wrapping mode, the element delta values shall **not** include the lengths of the "KL" for each element. The result is that each element delta shall point to the first byte of the value in the KLV that wraps an element.

Table A.7 – Clip wrapped delta entry array example for figure 6

	Field Name	Type	Meaning	Use
	NDE	UInt32	Number of delta entries	4
	Length	UInt32	Length of each delta entry	6
Picture Delta Entry	PosTableIndex	Int8	0= No reordering +ve = PosTable Index	0
	Slice	UInt8	Slice number in IndexEntry	0
	Element Delta	UInt32	Delta from start of slice to this Element	0
Sound Delta Entry	PosTableIndex	Int8	0= No reordering +ve = PosTable Index	0
	Slice	UInt8	Slice number in IndexEntry	1
	Element Delta	UInt32	Delta from start of slice to this Element	0
Sound Delta Entry	PosTableIndex	Int8	0= No reordering +ve = PosTable Index	0
	Slice	UInt8	Slice number in IndexEntry	2
	Element Delta	UInt32	Delta from start of slice to this Element	0
Data Delta Entry	PosTableIndex	Int8	0= No reordering +ve = PosTable Index	0
	Slice	UInt8	Slice number in IndexEntry	3
	Element Delta	UInt32	Delta from start of slice to this Element	0

Table A.8 – Clip wrapped index entry array example for figure 6

	N	Field Name	Type	Meaning	Use
	1	NIE	UInt32	Number of index entries	=number of frames
	1	Length	UInt32	Length of each index array entry	
One Index Entry for every frame	NIE	Temporal Offset	Int8	Offset in edit units from Display Order to Coded Order	0
		Anchor Offset	Int8	Offset in edit units to previous Anchor Frame. The value is zero if this is an anchor frame.	0
		Flags	EditUnitFlag	Flags for this Edit Unit Bit 7: Random Access Bit 6: Sequence Header Bit 5: forward prediction flag Bit 4: backward prediction flag e.g. 00== I frame (no prediction) 10== P frame(forward prediction from previous frame) 01== B frame (backward prediction from future frame) 11== B frame (forward & backward prediction) Bits 0-3: reserved [RP210 Flags to indicate coding of elements in this edit unit]	80h
		Stream Offset	UInt64	Offset in bytes from the first KLV element in this Edit Unit within the Essence Container Stream	Offset from the first byte of the value of the Uncompressed Picture KLV for the first frame to the first byte of the stored data for this Uncompressed Picture frame in Figure 6
		SliceOffset	UInt32	The offset in bytes from the Stream Offset to the start of slice 1.	Offset from the first byte of the stored data for this Uncompressed Picture frame to the first byte of the corresponding audio sample in the first Sound Element in Figure 6
		SliceOffset	UInt32	The offset in bytes from the Stream Offset to the start of slice 2	Offset from the first byte of the stored data for this Uncompressed Picture frame to the first byte of the corresponding audio sample in the second Sound Element in Figure 6
		SliceOffset	UInt32	The offset in bytes from the Stream Offset to the start of slice 3	Offset from the first byte of the stored data for this Uncompressed Picture frame to the first byte of the corresponding audio sample in the Data Element in Figure 6

A.5 Line wrapping

Index tables are based on edit units. the line-wrapping mode contains content packages that are smaller than one edit unit. In order to index each frame, an index table is built that points to the first byte of the key for the first KLV containing picture data (i.e., where byte 15 == 01h). It is possible that all the frames will be of a different length, as in the example below. If all frames are of the same length then an index table as described in A.1 can be built.

Table A.9 – Line wrapped index table segment set example

Item Name	Req ?	Meaning	Use
Index Table Segment	Req	An Index Table Segment set	See MXF Format Specification
Length	Req	Set Length	See MXF Format Specification
Instance ID	Req	Unique ID of this instance	See MXF Format Specification
Index Edit Rate	Req	Edit Rate copied from the tracks of the Essence Container	See MXF Format Specification
Index Start Position	Req	The first editable unit indexed by this Index Table segment measured in File Package Edit Units	
Index Duration	Req	Time duration of this table segment measured in Edit Units of the referenced Package	
Edit Unit Byte Count	D/Req	Defines the byte count of each and every Edit Unit. A value of 0 indicates that the byte count of Edit Units is defined in the Index Entry Array	0 unless the total length of each and every Line Wrapped Content Package is of constant size. In this example the value is 0.
IndexSID	D/Req	Stream Identifier (SID) of Index Table	See MXF Format Specification
BodySID	Req	Stream Identifier (SID) of the indexed Essence Container	See MXF Format Specification
Slice Count	D/Req	Number of slices minus 1 (NSL)	0
Delta Entry Array	Opt	Map Elements onto Slices	Table A.10
Index Entry Array	D/Req	Index from Edit Unit number to stream offset	Table A.11

The delta entry array shall contain a single entry for each picture in the generic container.

In this example, we will assume that each picture is of variable length so that an index entry array is required.

Table A. 10 – Line wrapped delta entry array example

	Field Name	Type	Meaning	Use
	NDE	UInt32	Number of delta entries	4
	Length	UInt32	Length of each delta entry	6
Picture Delta Entry	PosTableIndex	Int8	0= No reordering +ve = PosTable Index	0
	Slice	UInt8	Slice number in IndexEntry	0
	Element Delta	UInt32	Delta from start of slice to this Element	0

Table A.11 – Line wrapped index entry array example

	N	Field Name	Type	Meaning	Use
	1	NIE	UInt32	Number of index entries	=number of frames
	1	Length	UInt32	Length of each index array entry	
One Index Entry for every frame	NIE	Temporal Offset	Int8	Offset in edit units from Display Order to Coded Order	0
		Anchor Offset	Int8	Offset in edit units to previous Anchor Frame. The value is zero if this is an anchor frame.	0
		Flags	EditUnitFlag	Flags for this Edit Unit Bit 7: Random Access Bit 6: Sequence Header Bit 5: forward prediction flag Bit 4: backward prediction flag e.g. 00== I frame (no prediction) 10== P frame(forward prediction from previous frame) 01== B frame (backward prediction from future frame) 11== B frame (forward & backward prediction) Bits 0-3: reserved [RP210 Flags to indicate coding of elements in this edit unit]	80h
		Stream Offset	UInt64	Offset in bytes from the first KLV element in this Edit Unit within the Essence Container Stream	Offset from the first byte of the key of the KLV for the first frame to the first byte of the Key of the KLV for the System Item that starts this frame.

Annex B (informative)
Bibliography

ANSI/SMPTE 259M-1997, Television — 10-Bit 4:2:2 Component and 4fsc Composite Digital Signals — Serial Digital Interface

ANSI/SMPTE 267M-1995, Television — Bit-Parallel Digital Interface — Component Video Signal 4:2:2 16x9 Aspect Ratio

ANSI/SMPTE 298M-1997, Television — Universal Labels for Unique Identification of Digital Data

SMPTE 125M-1995, Television — Component Video Signal 4:2:2 — Bit-Parallel Digital Interface

SMPTE 274M-2005, Television — 1920 x 1080 Image Sample Structure, Digital Representation and Digital Timing Reference Sequences for Multiple Picture Rates

SMPTE 293M-2003, Television — 720 x 483 Active Line at 59.94-Hz Progressive Scan Production — Digital Representation

SMPTE 296M-2001, Television — 1280 x 720 Progressive Image Sample Structure — Analog and Digital Representation and Analog Interface

SMPTE 382M, Television — Material Exchange Format (MXF) — Mapping AES3 and Broadcast Wave Audio into the MXF Generic Container

SMPTE EG 41-2004, Material Exchange Format (MXF) — Engineering Guideline

ITU-R BT.470-7 (02/05), Conventional Analogue Television Systems

ITU-R BT.601-5 (10/95), Studio Encoding Parameters of Digital Television for Standard 4:3 and Wide-Screen 16:9 Aspect Ratios

ITU-R BT.656-4 (02/98), Interfaces for Digital Component Video Signals in 525 Line and 625 Line Television Systems Operating at the 4:2:2 Level of Recommendation ITU-R BT.601 (Part A)

ITU-R BT.709-5 (04/02), Parameter Values for the HDTV Standards for Production and International Programme Exchange

ITU-R BT.1358 (02/98), Studio Parameters of 625 and 525 Line Progressive Scan Television Systems

ITU-R BT.1361 (02/98), Worldwide Unified Colorimetry and Related Characteristics of Future Television and Imaging Systems