SMPTE REGISTERED DISCLOSURE DOCUMENT

RDD 9-2009
Revision of RDD 9-2006

MXF Interoperability Specification of Sony MPEG Long GOP Products

The attached document is a Registered Disclosure Document prepared by the proponent identified below. It has been examined by the appropriate SMPTE Technology Committee and is believed to contain adequate information to satisfy the objectives defined in the Scope, and to be technically consistent.

This document is NOT a Standard, Recommended Practice or Engineering Guideline, and does NOT imply a finding or representation of the Society.

Errors in this document should be reported to the proponent identified below, with a copy to eng@smpte.org. All other inquiries in respect of this document, including inquiries as to intellectual property requirements that may be attached to use of the disclosed technology, should be addressed to the proponent identified below.

Proponent contact information:

Hiroshi Nakano
Sony Corporation, B2B
4-14-1 Asahicho, Atsugi
Kanagawa, 243-0014
Japan

Email: hiroshi.nakano@jp.sony.com

Approved
June 22, 2009

Table of Contents

Page

1 Scope

This document specifies the mapping of MPEG-2 Picture (ES) compatible with MPEG stream and AES3 audio into the MXF Generic Container. The MXF files created according to the details of this specification comply with the MXF specifications defined in the normative references. In conjunction with the referenced Standards, this RDD is intended to provide sufficient information to enable a developer to construct MXF files that will be compatible with Sony MPEG Long GOP products.

2 Normative References

SMPTE 326M-2000, Television — SDTI Content Package Format (SDTI-CP)

SMPTE 331M-2004, Television — Element and Metadata Definitions for the SDTI-CP

SMPTE 377-1-2009, Material Exchange Format (MXF) — File Format Specification

SMPTE 378M-2004, Television — Material Exchange Format (MXF) — Operational Pattern 1a (Single Item, Single Package)

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

SMPTE 381M-2005, Television — Material Exchange Format (MXF) — Mapping MPEG Streams into the MXF Generic Container

SMPTE 382M-2007, Material Exchange Format — Mapping AES3 and Broadcast Wave Audio into the MXF Generic Container

SMPTE 385M-2004, Television — Material Exchange Format (MXF) — Mapping SDTI-CP Essence and Metadata into the MXF Generic Container

SMPTE 400M-2004, Television — SMPTE Labels Structure

3 Introduction

The MXF Generic Container is a streamable Essence Container that can be placed on any suitable transport and stored. SMPTE 379M defines the MXF Generic Container as the native Essence Container in MXF files. SMPTE 381M defines how MPEG streams, as identified by an ISO 13818-1 stream id value, can be mapped in the MXF Generic Container. SMPTE 382M defines how AES3 and Broadcast Wave Audio can be mapped in the MXF Generic Container. SMPTE 385M defines the System Item that is compatible with SMPTE 326M (SDTI-CP) and also defines how SDTI-CP essence and metadata can be used in the MXF Generic Container. This document specifies the mapping of MPEG Picture (ES) compatible with MPEG stream and AES3 audio into the MXF Generic Container. This document also specifies the MXF file format which includes unique identifiers, operation pattern, partitions, index table segments and RIP. The common basic structure is described in this document

4 Outline of MXF File Structure for this Mapping

Figure 1 shows the outline of the MXF file structure. The file consists of a Header Partition, segmented Body partition(s), a Footer Partition and a Random Index pack. Picture, Sound and System Items are mapped into the Essence Container and placed in each Body Partitions. The Data Item is optional. Because of MPEG Long GOP structure of Picture Item, segmented Index Tables are used together with Random Index pack. More detailed explanation can be found in Section 7 (Application Issues).

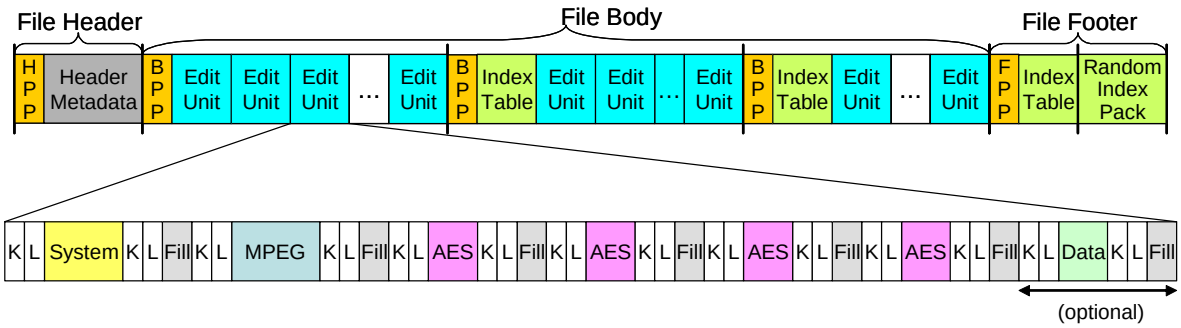


Figure 1 – Outline of MXF File

- **HPP**: Header Partition Pack
- **BPP**: Body Partition Pack
- **FPP**: Footer Partition Pack
- **Fill**[1]: The length of the KLV Fill item should be required to align to a KAG boundary.

Some of the aspects of this structure are shown below.

- It is only necessary to include one index table segment for each Body Partition period on the sender side.
- It is easy to perform the function “Play while receiving file” on the receiver side.
- It is easy to pick extract a “Partial file”.
- A list of constraints for this file structure is given in Table 1.

Table 1 – Constraints for Sony MPEG Stream Products

Item	Constraints
Operational Pattern	1a - *Origin* and *Duration* are used for express the GOP precharge and overrun.
Wrapping (Interleaving)	Frame by Frame (coded order)
KAG[1] size	512
Partitions	Multiple Body Partitions - At least 1GOP and up to approximately 10 seconds body partition - There are an integer number of GOPs per partition (>=1). - Audio and other Essence have the same extent as Video Essence
Index Table	Index Tables are divided into Index Table Segments spanning the same time interval as the Essence in each Partition. - Index the essence of *previous* partition, therefore there is no Index Table Segment in the first Body Partition, and the Footer Partition has the last one.
System Item	Compliant to 326M and 385M, use for the Frame by Frame Timecode, UMID
Video packetization	Compliant to 381M, MPEG-2 ES

GOP structure	Max 15 frames/GOP (N<=15, M<=3, open GOP / closed GOP) // 1920x1080i
	Max 12 frames/GOP (N<=12, M<=3, open GOP / closed GOP) // 1280x720p
	Variable length GOPs are permitted.

Audio sampling	48KHz locked
Audio packetization	Compliant to 382M, AES3, 1ch/element (min 2 to max 8 channels)
Data Item (optional)	Compliant to 331M and 385M, Optionally used for VBI data and Ancillary packet
Timecode	System Item and Header Metadata

[1] Refer to 7.1 Application of the KLV Fill Item

5 MPEG Picture Data and AES3 Data Mapping

The mapping of MPEG Picture (ES) data is as defined in SMPTE 381M. The mapping of AES3 digital audio data is defined by SMPTE 382M. This specification uses frame-based wrapping as defined by SMPTE 379M. The System Item is defined by SMPTE 326M, and mapped into the MXF by SMPTE 385M The order of Items in each Edit Unit is System, Picture, Sound and Data. (where the Data item is optional)

5.1 Frame-Based Wrappings

This document requires the use of frame-based wrapping as defined by SMPTE 379M Section 5.1.

In the case of audio locked to video at 25 (or 50) content packages per second, each element will contain the same number of samples, for example 1920 (or 960).

In the case of audio locked to video at 30*1000/1001 content packages per second, the number of samples in each element will vary to maintain a correct aggregate rate. Typically the number of samples varies according to a 5-frame sequence, 1602, 1601, 1602, 1601, and 1602.

In the case of audio locked to video at 60*1000/1001 content packages per second, the number of samples in each element will vary to maintain a correct aggregate rate. Typically the number of samples varies according to a 5-frame sequence, 801, 801, 800, 801, and 801.

The number of samples in each content package is calculated from the Length field of the surrounding KLV packet, divided by the value of the BlockAlign property of the AES3 Audio Essence Descriptor.

In the case of audio locked to video at 24*1000/1001 content packages per second, each element will contain the same number of samples, for example 2002.

An arrangement of System, Picture, and Sound Items in a frame-based wrapping is shown in Figure 2. It shows the case of 4 channels AES3 audio.

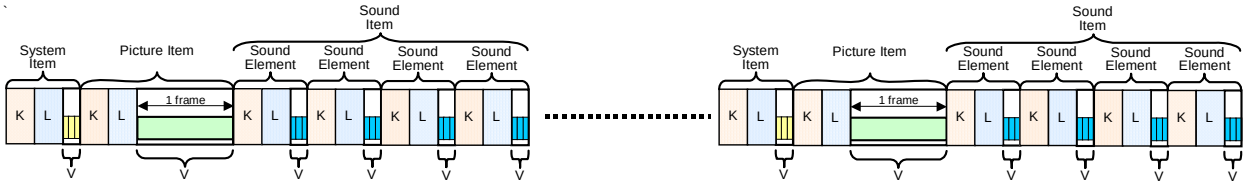


Figure 2 – Frame Wrapping of System, Picture and Sound Elements

5.2 System Item Mapping

The System Item in each Edit Unit consists of System Metadata Pack and a Package Metadata set.

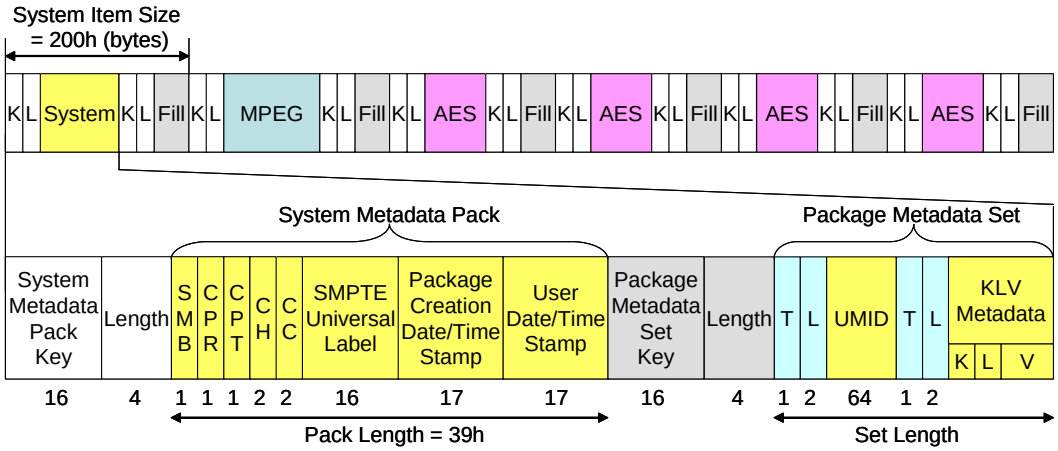


Figure 3 – Mapping of System Item in an Edit Unit

The mapping of System Item data complies with that defined in SMPTE 385M. The System Metadata Pack conforms to 326M and the items in the Package Metadata Set are defined by 331M.

System Metadata Pack Key: 06 0E 2B 34 02 05 01 01 0D 01 03 01 04 01 01 00

Package Metadata Set Key: 06 0E 2B 34 02 43 01 01 0D 01 03 01 04 01 02 xx

The 16th byte of the Package Metadata Set Key represents the number of metadata blocks in the Element.

The LTC data is carried in User Date/Time Stamp, and the UMID and the KLV Metadata are carried in Package Metadata Set.

UMID (basic or extended)	Tag = 83h (as defined in SMPTE 331M)
KLV Metadata	Tag = 88h (as defined in SMPTE 331M)

5.3 Picture Item Mapping

The Picture Item for this mapping specification is Frame Wrapped and the data length of each Picture Element in an Edit Unit will be variable value.

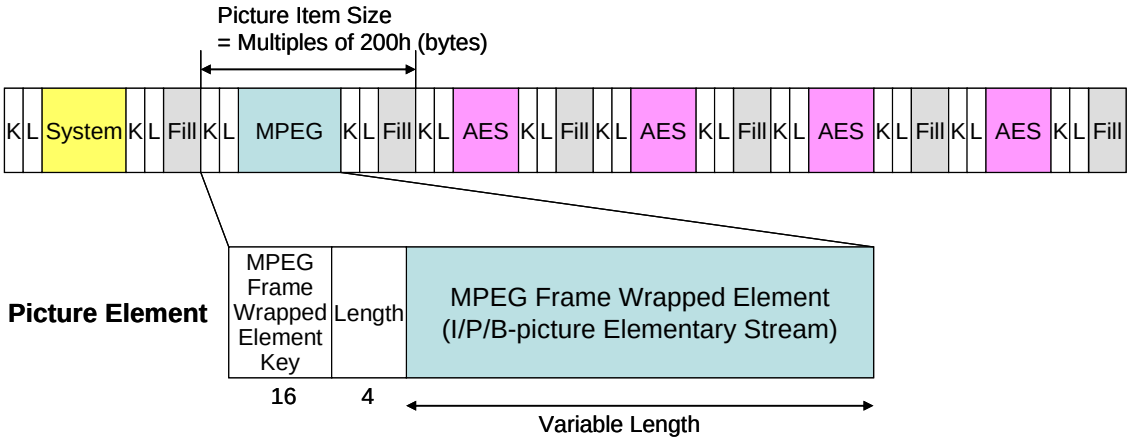


Figure 4 – Mapping of Picture Item in Edit Unit

5.3.1 MPEG Picture Element Key

The values of the first 12 bytes of the essence element Key are defined in SMPTE 379M, MXF Generic Container Format. The picture element is defined by SMPTE 381M, Mapping MPEG streams into the MXF Generic Container Format. This defines byte 13 with the value of 15h. The values of the last four bytes of the essence Element Key are given in Table 2.

Table 2 – Key Value for the MPEG Picture Element

Byte No.	Description	Value (hex)	Meaning
1-12	Specified by the MXF Generic Container Specification SMPTE 379M		
13	Item Type Identifier	15h	Picture Item
14	Essence Element Count	01h	Count of Elements in this Item
15	Essence Element Type	05h	Frame Wrapped Picture Element
16	Essence Element Number	00h	Element Unique Number

5.3.1.1 MPEG Picture Element Number — Byte 16

This is a number used as an index to identify this instance of the Element within the Item. It does not change within any instance in the Generic Container.

5.3.2 MPEG Picture Element Length

The length field of the KLV coded Element is 4 bytes BER long-form encoded (i.e. 83h.xx.yy.zz) for Frame wrapping.

5.3.3 MPEG Picture Element Value

The MPEG-2 picture element complies with SMPTE 381M. The maximum frame number per GOP is fifteen frames, and both open and closed GOP modes are supported.[2]

5.4 AES3 Sound Item Mapping

The AES3 Sound element complies with 382M, Mapping AES3 and Broadcast Wave Audio into the MXF Generic Container. Sound Item for this mapping specification is Frame Wrapped.

[2] Refer to Table 1.

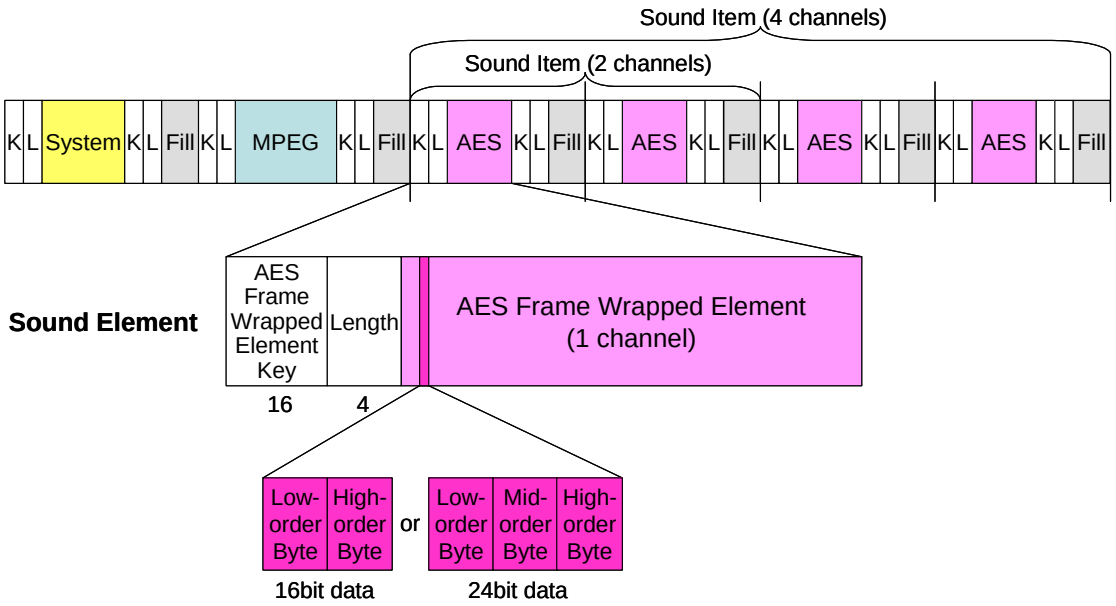


Figure 5 – Mapping of Sound Item

5.4.1 AES3 Sound Element Key

The values of the first 12 bytes of the essence element Key are defined in SMPTE 379M, MXF Generic Container Format. The values of the last four bytes of the essence Element Key are given in Table 3.

Table 3 – Key Value for the AES3 Sound Element

Byte No.	Description	Value (hex)	Meaning
1-12	Specified by the MXF Generic Container Specification SMPTE 379M		
13	Item Type Identifier	16h	GC Sound Item
14	Essence Element Count	02, 04 or 08h	Count of Sound Elements in this Generic Container
15	Essence Element Type	03h	AES Frame Wrapped Element as listed in SMPTE RP 224
16	Essence Element Number	00, 01,02,03, 04,05,06 or 07h	Element Unique Number

5.4.1.1 AES3 Sound Element Count — Byte 14

This is a count of the number of Elements in the Sound Items in the Generic Container. The count of the Sound Elements for this mapping specification may be 2, 4 or 8 according to the count of available audio channels.

5.4.1.2 AES3 Sound Element Number — Byte 16

This is a number used as an index to identify this instance of the Element Type within the Item. Each Element within an Item has a unique value in sequence from 00h upwards as defined by SMPTE 379M, which remain constant throughout any instance of a Generic Container.

5.4.2 AES3 Sound Element Length

The length field of the KLV coded Element is 4 bytes BER long-form encoded (i.e. 83h.xx.yy.zz) for Frame wrapping.

5.4.3 AES3 Sound Element Value

In a Frame Wrapped file, the Start Position of the first sample of AES3 audio data should be the same as the Start Position of the synchronized Picture Element. Note that the order of Frame Wrapped Picture Item of MPEG Picture Element is not display order, but coding order, i.e. I2 B0 B1 P5 B3 B4 P8 B6 B7…,. Therefore, Sound and Picture Elements are not necessarily synchronized within each edit unit.

5.5 Data Item Mapping (Optional)

Data Item in each Edit Unit may consist of two kinds of Data Element; i.e., VBI line element and Ancillary Data Packet element.

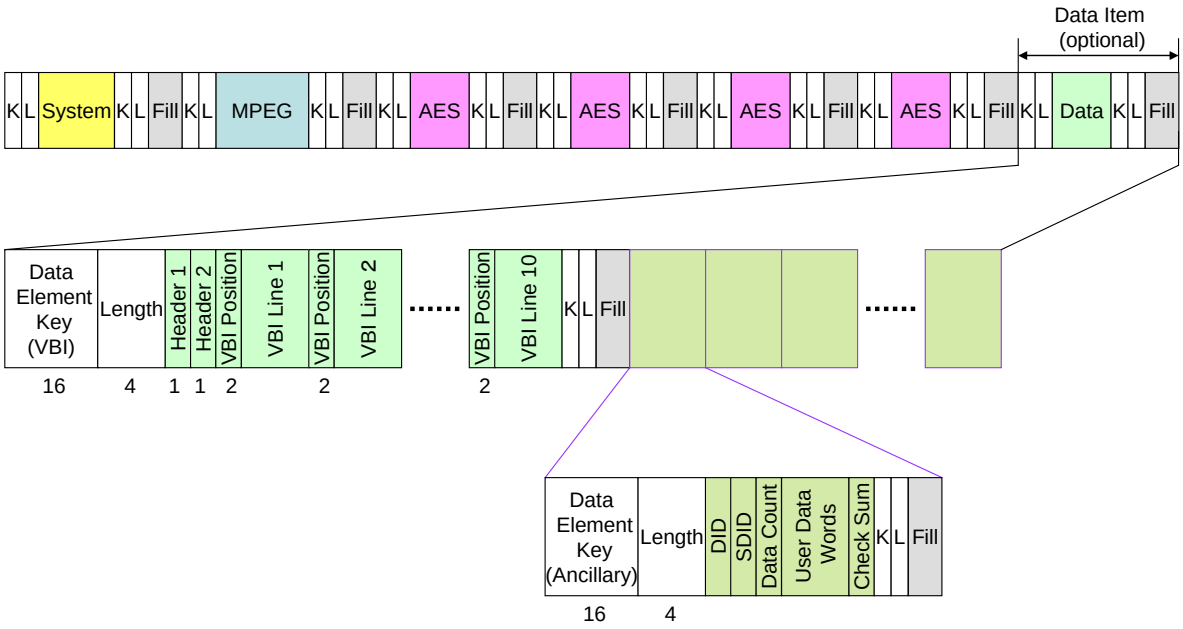


Figure 6 – An Example of Data Item in an Edit Unit

The Data Item complies with SMPTE 331M VBI line and Ancillary data packet. The Data Element Keys for VBI line and Ancillary data packet are as given below:

VBI line element Key: 06 0E 2B 34 01 02 01 01 0D 01 03 01 07 01 20 00

Ancillary data packet element Key: 06 0E 2B 34 01 02 01 01 0D 01 03 01 07 xx 21 00

The 14th byte of the Ancillary data packet element Key represents the number of essence Elements. Other wrapping of VBI and Ancillary data packet are permitted.

5.6 Temporal Reordering

In the case of MPEG long GOP video, the compressed video pictures are reordered from their display order according to the MPEG specification. This reordering is applied only to the video elements.[3]

[3] Refer to Figure 11 in Section 7.3.5

6 SMPTE Labels for Essence Container Identification

The values for the MPEG Video Essence Container UL are given in Table 4.

Table 4 – Specification of the MPEG Video Essence Container Label

Byte No.	Description	Value (hex)	Meaning
1-12	Specified by the MXF Generic Container Specification SMPTE 379M		
13	Essence Container Kind	02h	MXF Generic Container
14	Mapping Kind	04h	MPEG ES as listed in SMPTE RP 224
15	Locally defined	60h	1110xxxx : ITU-T Rec. H.262 | ISO/IEC 13818-2 or ISO/IEC 11172-2 video stream number xxxx ISO13818-1 stream_id bits 6..0 as described in SMPTE 381M
16	Locally defined	01h	MPEG ES, Frame Wrapping as defined in SMPTE RP 224

The AES3 Audio specific values for the Essence Container UL are given in Table 5.

Table 5 – Specification of the AES3 Audio Essence Container Label

Byte No.	Description	Value (hex)	Meaning
1-12	Specified by the MXF Generic Container Specification SMPTE 379M		
13	Essence Container Kind	02h	MXF Generic Container
14	Mapping Kind	06h	AES-BWAV as listed in SMPTE RP 224
15	Content Kind	03h	AES Frame Wrapped Element as listed in SMPTE RP 224
16	Reserved	00h	

These SMPTE Labels are the values of the ‘Essence Container’ property used in the Partition Pack, Preface Set and appropriate File Descriptor Sets.

7 SMPTE Labels for Essence Coding Identification

The values for the Picture Essence Coding UL are given in Table 6.

Table 6 – Specification of the Picture Essence Coding Label

Byte No.	Description	Value (hex)	Meaning
1-8	Specified by the SMPTE Labels Structure Specification SMPTE 400M		
9	Parametric	04h	Node used to define parametric data
10	Picture Essence	01h	Identifies picture essence coding
11	Picture Coding Characteristics	02h	Identifies picture coding characteristics
12	Compressed Picture Coding	02h	Identifies compressed picture coding
13	MPEG picture coding	01h	Identifies MPEG picture coding
14	MPEG-2 Profile and Label	03h 04h 05h	MPEG-2 MP@HL MPEG-2 422P@HL MPEG-2 MP@H14
15	Long GOP Coding	03h	Identifies long gop coding
16	Reserved	00h	

The Picture Essence Coding UL is used in the Generic Picture Essence Descriptor. This UL is listed in the SMPTE labels registry (SMPTE RP 224).

The Generic Sound Essence Descriptor does not include the Sound Essence Coding property. MXF decoders should assume the value of the property is UL of SMPTE 382M uncompressed sound coding (i.e. 06.0E.2B.34.04.01.01.0A.04.02.02.01.01.00.00.00).

8 Application Issues

8.1 Application of the KLV Fill Item

Within any MXF partition containing an Essence Container with this mapping specification, the KAG value defined in the Partition Pack has the value of 512 (02.00h) and the first byte of the Key of the first Element of each Item is aligned to the KLV alignment grid of that partition.

For each Item in a Content Package, the length of the KLV Fill item should be the minimum required to align to a KAG boundary.

Where possible, any immediately preceding partition should align the start of each MXF partition containing an Essence Container with this mapping specification to a byte offset that is an integer multiple of the defined KAG relative to the start of the Header Partition Pack.

The length field of the KLV fill item is 4 bytes, BER long-form encoded (i.e., 83h.xxh.yyh.zzh).

8.2 Application of Body Partition

The Essence Container in a MXF file conforming to this mapping specification is divided into Body Partitions as shown in Figure 7. Each Body partition contains at least 1 GOP and up to 10 seconds of Picture and Sound Essence. The MPEG2 essence element in the Essence Container of each partition is aligned with the GOP boundary in all cases.

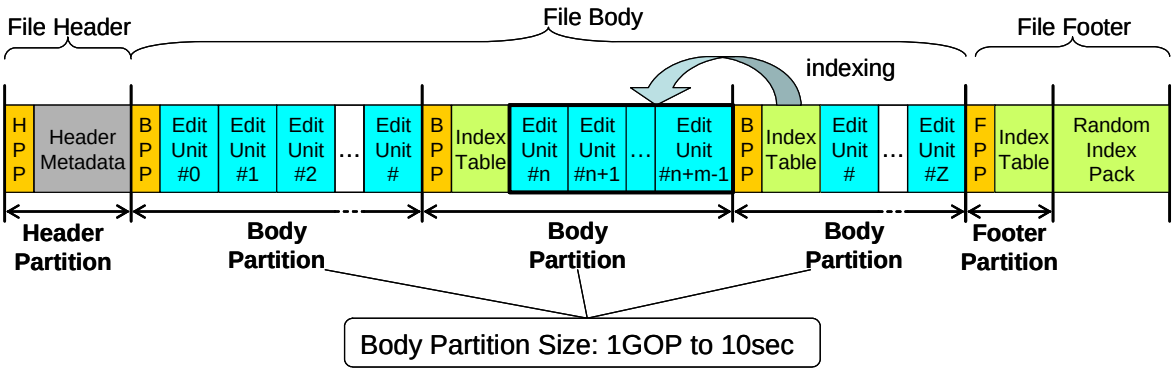


Figure 7 – File Structure and Body Partitions

The purpose of divided Body Partitions is to adopt Index Table Segments in each partition. Each Index Table Segment is carried in the Partition immediately following the Essence it indexes. Random Index Pack is placed in file footer in order to index position of each Body Partition.

8.3 Application of Index Table for Frame Wrapped MPEG Picture and AES Sound Essence

8.3.1 Essence Container and Index Table

Long GOP essence requires the use of Index Table segments. Index Entry and Delta Entry arrays are required. One Index Table Segment is placed in each Body Partition except the first Body partition immediately following Header Partition. The definition of the Index Table format is given in the MXF File Format Specification (SMPTE 377-1).

Note: The limit of 10 seconds for each partition ensures that only a single Index Table Segment is required to index the long GOP essence in each partition.

As noted in the previous section, each Index Table indexes the Essence Container in the previous Body Partition. Owing to this Index Table arrangement, real time creation of MXF file can be performed without buffering each Body Partition. In this mapping specification Picture Essence is VBE (Variable Bytes per Element) and Sound Essence is CBE (Constant Bytes per Element).

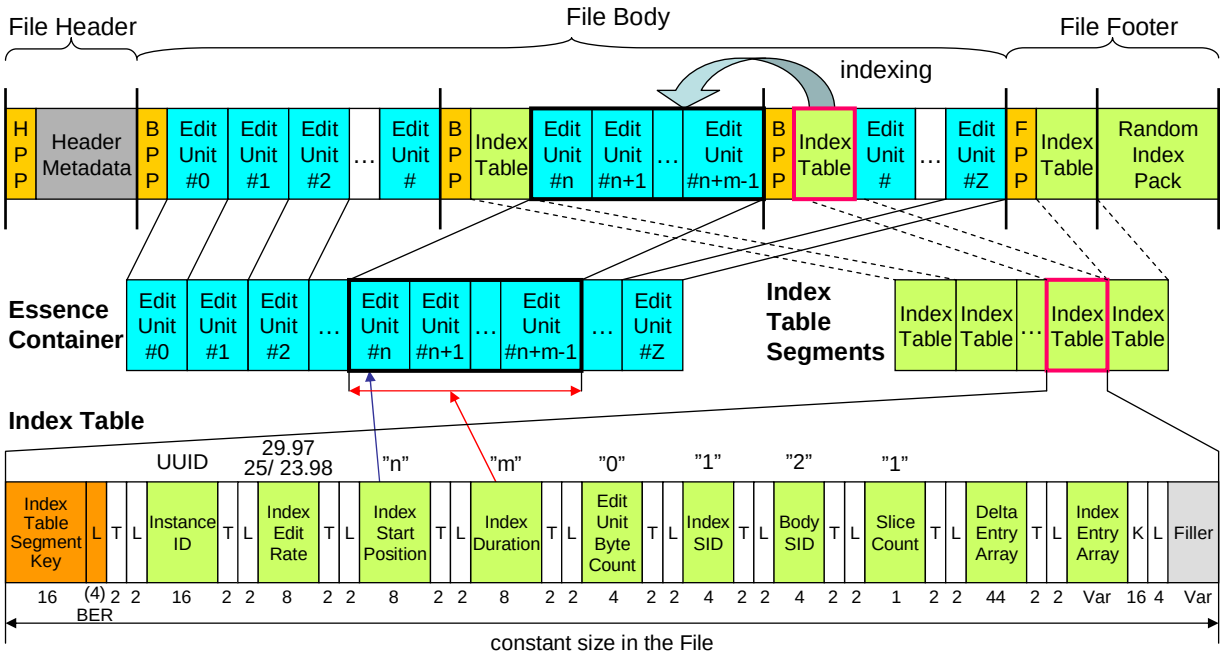


Figure 8 – Essence Container and Index Table Segments

8.3.2 Index Table Items

The Index Table Segment is constructed as Table 7.

Table 7 – Index Table Segment Set Example

Item Name	Req ?	Meaning	Use for this mapping
Index Table Segment Key	Req	An Index Table Segment set	
Length	Req	Set Length	
Instance ID	Req	Unique ID of this instance	
Index Edit Rate	Req	Edit Rate copied from the tracks of the Essence Container	{60000,1001}, {50,1}, {30000,1001}, {25,1} or {24000, 1001}
Index Start Position	Req	The first editable unit indexed by this Index Table segment measured in File Package Edit Units	This sets the temporal start point for an Index Table segment
Index Duration	Req	Time duration of this table segment measured in Edit Units of the referenced Package	Combined with Start Position, allows an application to determine if this Index Table Segment spans a particular Position value
Edit Unit Byte Count	D/Req	Defines the byte count of each and every Edit Unit. A value of 0 defines the byte count of Edit Units is only given in the Index Entry Array	Edit Unit length is variable, so the value is 0.
IndexSID	D/Req	Stream Identifier (SID) of Index Table	
BodySID	Req	Stream Identifier (SID) of the indexed	

Item Name	Req ?	Meaning	Use for this mapping
		Essence Container	
Slice Count	D/Req	Number of slices minus 1 (NSL)	1
PosTableCount	Opt	Number of PosTable Entries minus 1 NPE	Not Encoded
Delta Entry Array	Opt	Map Elements onto Slices	
Index Entry Array	Req	Index from Edit Unit number to stream offset	

An Index Table provides byte offset information within an Essence Container for a given time offset from the start of that Essence Container. If the Essence Container has interleaved data within in it, then extra mechanisms are provided for finding the offsets to the individual Essence Elements once the correct time offset is located. Each Index Entry provides the byte offset within the file for a given time offset measured in Edit Units.

To locate the individual elements within the Index Entry, the Delta Entries and Slice Offsets are required. The extent to which the Essence is indexed depends on an application. For many applications, simple indexing of the start of each edit unit will suffice. It is then up to the decoder within the application to find the start of each element by parsing.

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. However, in this mapping type, Picture Element is VBE so that an Index Entry array is required.

8.3.3 Delta Entry Array

The Slice mechanism is introduced to describe all Elements' offsets in each Edit Units effectively. Each Slice starts with a number of CBE elements and ends with a single VBE element (or end of the Unit). The start of Slice zero corresponds to the start of the Index Entry.

The Delta Entry Array contains an entry for every indexed Element in the Generic Container. The order of the elements in the Delta Entry Array matches the order of the Elements in the Generic Container.

Any Element that has minor sample variations (e.g. audio 5-frames sequence) is padded to a constant size if it is to be regarded as a CBE, otherwise it is a VBE element and uses the slice mechanism. The Essence Type of the Delta Entry is determined by inspection of the Key of the Essence (e.g. Picture, Sound, Fill etc.)

In this mapping type, there are two Slices in the Edit Unit.

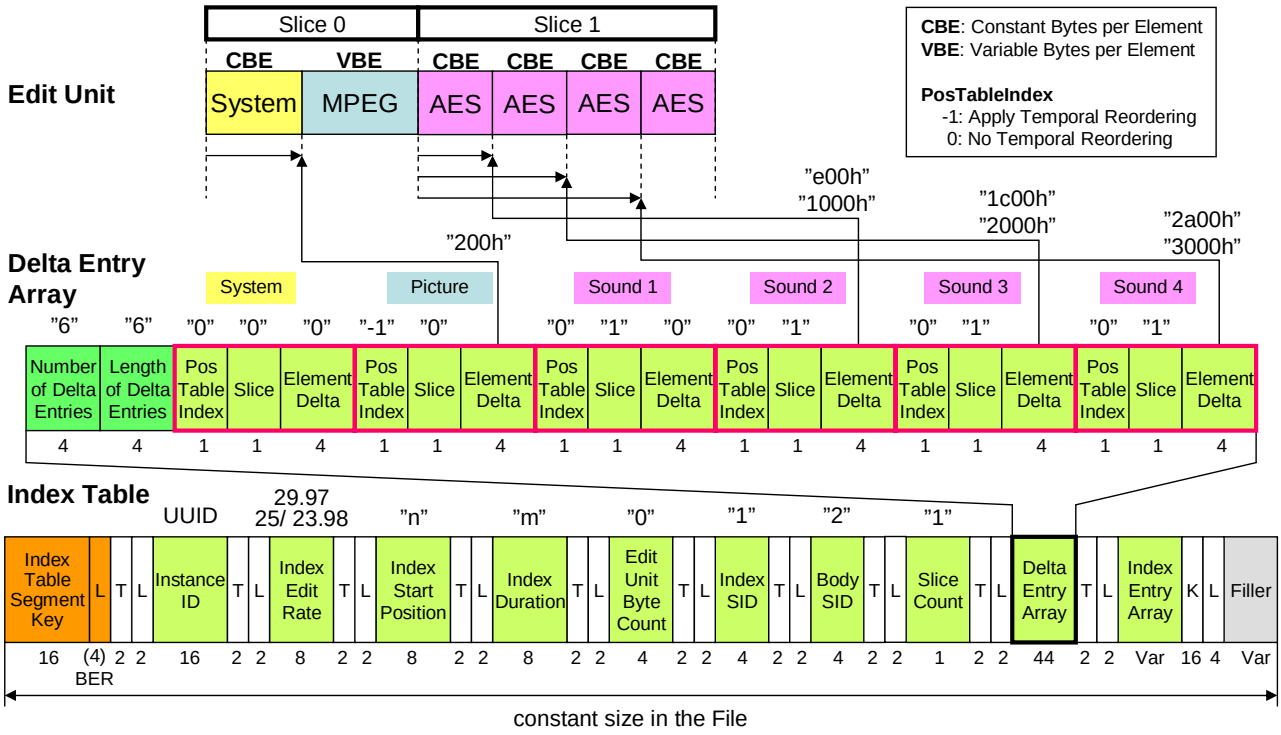


Figure 9 – Delta Entry Array

PosTableIndex is used to discover if this element has been temporally reordered or not. If the value is zero, there is no reordering and no Temporal offsetting for this element. If the value is negative then the Temporal Offset property of the IndexEntryArray is used to determine the difference between presentation order and storage order of the Indexed Element. In this mapping type, positive value is not used.

Slice value is the Slice number in the Edit Unit.

Element Delta is the byte offset from start of the Slice to this Element. Element Delta values include the lengths of the "KL" for each Element. The result is that each Element Delta points to the first byte of the Key in the KLV which wraps an element.

For this mapping specification, there are 4 or 6 or 10 Delta Entries (for System, Picture and 2/4/8 Sounds) defined as follows:

The example in Table 8 below is a Delta Entry Array designed to match Figure 9, which is the case of 16-bit and 4-channels audio.

Table 8 – Structure of Delta Entry Array

	Field Name	Type	Meaning	Use for this mapping
	NDE	UInt32	Number of delta entries	6
	Length	UInt32	Length of each delta entry	6
System Delta Entry	PosTableIndex	Int8	0= No temporal reordering -1 = Apply temporal reordering	0
	Slice	UInt8	Slice number in Index Entry	0
	Element Delta	UInt32	Delta from start of slice to this Element	0
Picture Delta Entry	PosTableIndex	Int8	0= No temporal reordering -1 = Apply temporal reordering	-1 (reordered Long GOP content)
	Slice	UInt8	Slice number in Index Entry	0
	Element Delta	UInt32	Delta from start of slice to this Element	200h
Sound 1 Delta Entry	PosTableIndex	Int8	0= No temporal reordering -1 = Apply temporal reordering	0
	Slice	UInt8	Slice number in Index Entry	1
	Element Delta	UInt32	Delta from start of slice to this Element	0
Sound 2 Delta Entry	PosTableIndex	Int8	0= No temporal reordering -1 = Apply temporal reordering	0
	Slice	UInt8	Slice number in Index Entry	1
	Element Delta	UInt32	Delta from start of slice to this Element	e00h (59.94i 29.97p) 1000h (50i, 25p, 23.98p)
Sound 3 Delta Entry	PosTableIndex	Int8	0= No temporal reordering -1 = Apply temporal reordering	0
	Slice	UInt8	Slice number in Index Entry	1
	Element Delta	UInt32	Delta from start of slice to this Element	1c00h (59.94i 29.97p) 2000h (50i, 25p, 23.98p)
Sound 4 Delta Entry	PosTableIndex	Int8	0= No temporal reordering -1 = Apply temporal reordering	0
	Slice	UInt8	Slice number in Index Entry	1
	Element Delta	UInt32	Delta from start of slice to this Element	2a00h (59.94i 29.97p) 3000h (50i, 25p, 23.98p)

8.3.4 Index Entry Array

The Index Entry provides the byte offset to the start of each edit unit within the Essence Container. Each Index Entry value marks the start of the Key for the KLV packet of the first element in each edit unit. The temporal distance between Index Entries within an Index Table Segment Set is one Edit Unit.

The Essence Type of the Index Entry is determined by inspection of the Key of the Essence (e.g. Picture, Sound, Fill etc.)

The Index Entry also provides some properties specific for Long GOP MPEG described in the next clause.

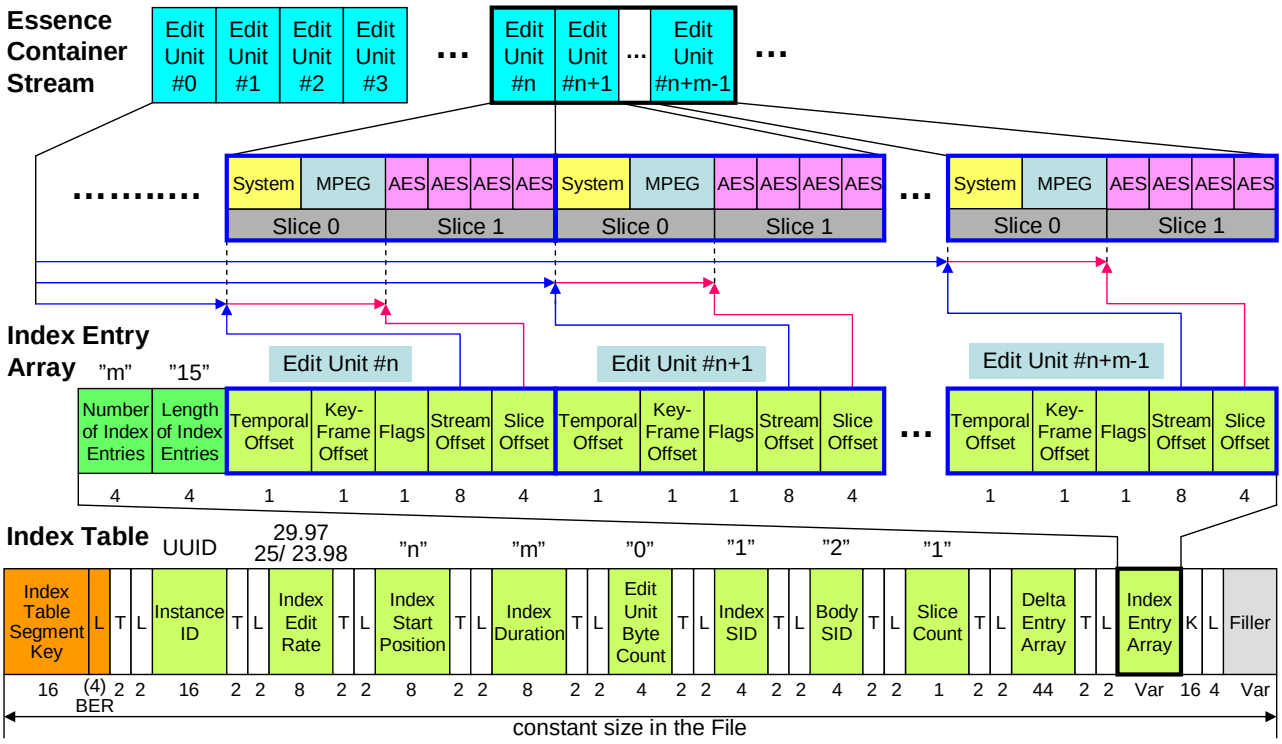


Figure 10 – Index Entry Array

Temporal Offset and **Key-Frame Offset** are properties specific for Long GOP MPEG. In that case, the compressed video pictures may be reordered from their display order according to the MPEG specification. This reordering is applied only to the video elements.

Flags represent Long GOP MPEG picture type.

Stream Offset is the byte offset from the beginning of the Essence Container Stream to the first KLV element in this Edit Unit.

Slice Offset values provide the byte offset within the Edit Unit to the start of any Slices. In this mapping type, there are 2 slices, so just 1 Slice Offset value in an Index Entry.

Each Index Entry value may have zero or more Slice Offset values that provide the byte offset within the edit unit to the end of any elements which are VBE.

Table 9 shows the descriptions of the various elements required in the Index Entry.

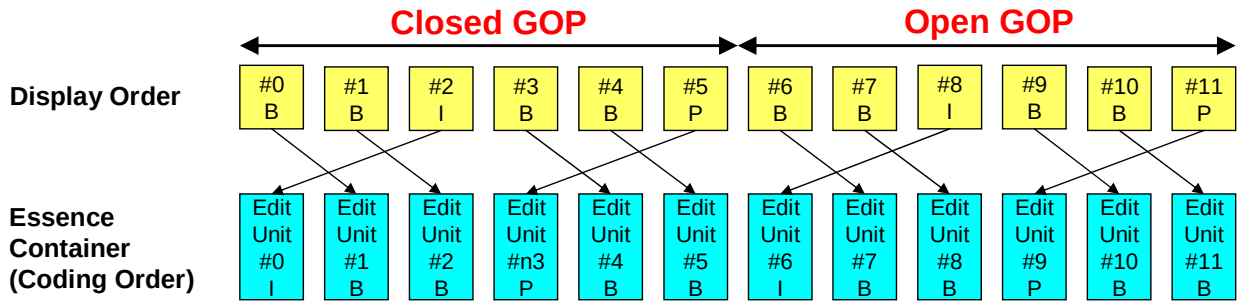
Table 9 – Index Entry Array Description

	N	Field Name	Type	Meaning	Use for this mapping
	1	NIE	UInt32	Number of index entries	Number of frames indexed by this Index Table segment
	1	Length	UInt32	Length of each index entry	15
One Index Entry for every frame	NIE	Temporal Offset	Int8	Offset in edit units from Display Order to Coded Order	According to picture type
		Key Offset	Int8	Offset in edit units to previous Key Frame. The value is zero if this is a Key frame.	According to picture type
		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 Bit 3: Offsets out of range Bit 2: Not used by MPEG Bits 0,1: MPEG Frame type	For Bits 5&4, naïve settings 00== I frame 10== P frame 11== B frame could be used if the strict setting value would be unknown.
		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 Data Element in this frame.
		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 there are 2 slices and NSL is set to 1
		PosTable	NPE *Rational	The fractional position offset from the start of the Content Package to the synchronized sample in the Content Package	Not encoded

8.3.5 Setting the Properties Specific for Long GOP MPEG

There are several properties in the Index Entry which have specific meanings for a Long GOP MPEG Index Table. These flags are correctly set according to Table 9.

Figure 11 shows an example of Temporal Offset, Key-Frame Offset, and Flags.



Temporal Offset	+1	+1	-2	+1	+1	-2	+1	+1	-2	+1	+1	-2
Key-Frame Offset	0	-1	-2	-3	-4	-5	0	-7	-8	-3	-4	-5
Flags	c0h	13h	13h	22h	33h	33h	40h	33h	33h	22h	33h	33h

Figure 11 – Long GOP MPEG Properties

Table 10 shows the Index entries for the first 6 frames of a Long GOP sequence in Figure 11.

The following example values are used in creating the table: (All lengths below are including Fill.)

- System item length is constant (= 200h bytes).
- Picture item length is variable (= 5e000h, f000h, f800h, 22000h, 10000h, and f000h in Edit Unit order).
- Sound item length is constant (= 3800h), which is the case of 4 channels.

Table 10 – Index Entry Array for the first 6 frames

	N	Field Name	Type	Value	Note
		NIE	UInt32	m	Number of Index Entries in this Index Table Segment
		Length	UInt32	15	Size of Index Entry
	Index Entry[0] contains Index data for I_2, and a Temporal offset from B_0 to Index Entry[1]				
0	Index Entry[0]	Temporal Offset	Int8	1	B_0 index data is stored in Index Entry[1]
		Key Frame Offset	Int8	0	The Key frame is IndexEntry[0]
		Flags	EditUnit Flag	c0h	I frame – no prediction, sequence_header & random access point
		Stream Offset	UInt64	0h	Offset of the 1st Stored Edit Unit in Essence Container – I_2
		Slice Offset	UInt32	5e200h	sizeof(System Item) + sizeof(I_2 Picture Item)
	Index Entry[1] contains Index data for B_0, and a Temporal offset from B_1 to Index Entry[2]				
1	Index Entry[1]	Temporal Offset	Int8	1	B_1 index data is stored in Index Entry[2]
		Key Frame Offset	Int8	-1	The Key frame is IndexEntry[0] and 0-1= -1
		Flags	EditUnit Flag	13h	Closed GOP B frame – backward prediction
		Stream Offset	UInt64	61a00h	Offset of the 2nd Stored Edit Unit in Essence Container – B_0
		Slice Offset	UInt32	f200h	sizeof(System Item) + sizeof(B_0 Picture Item)
	Index Entry[2] contains Index data for B_1, and a Temporal offset from I_2 to Index Entry[0]				
2	Index Entry[2]	Temporal Offset	Int8	-2	I_2 index data is stored in Index Entry[0]
		Key Frame Offset	Int8	-2	The Key frame is IndexEntry[0] and 0-2= -2
		Flags	EditUnit Flag	13h	Closed GOP B frame – backward prediction
		Stream Offset	UInt64	74400h	Offset of the 3rd Stored Edit Unit in Essence Container – B_1
		Slice Offset	UInt32	fa00h	sizeof(System Item) + sizeof(B_1 Picture Item)
	Index Entry[3] contains Index data for P_5, and a Temporal offset from B_3 to Index Entry[4]				
3	Index Entry[3]	Temporal Offset	Int8	1	B_3 index data is stored in Index Entry[4]
		Key Frame Offset	Int8	-3	The Key frame is IndexEntry[0] and 0-3= -3
		Flags	EditUnit Flag	22h	P frame – forward prediction
		Stream Offset	UInt64	87600h	Offset of the 4th Stored Edit Unit in Essence Container – P_5
		Slice Offset	UInt32	22200h	sizeof(System Item) + sizeof(P_5 Picture Item)
	Index Entry[4] contains Index data for B_3, and a Temporal offset from B_4 to Index Entry[5]				
4	Index Entry[4]	Temporal Offset	Int8	1	B_4 index data is stored in Index Entry[5]
		Key Frame Offset	Int8	-4	The Key frame is IndexEntry[0] and 0-4= -4
		Flags	EditUnit Flag	33h	B frame – bidirectional prediction
		Stream Offset	UInt64	ad000h	Offset of the 5th Stored Edit Unit in Essence Container – B_3
		SliceOffset	UInt32	10200h	sizeof(System Item) + sizeof(B_3 Picture Item)
	Index Entry[5] contains Index data for B_4, and a Temporal offset from P_5 to Index Entry[3]				
5	Index Entry[5]	Temporal Offset	Int8	-2	P_5 index data is stored in Index Entry[3]
		Key Frame Offset	Int8	-5	The Key frame is IndexEntry[0] and 0-5= -5
		Flags	EditUnit Flag	33h	B frame – bidirectional prediction
		Stream Offset	UInt64	c0a00h	Offset of the 6th Stored Edit Unit in Essence Container – B_4
		SliceOffset	UInt32	f200h	sizeof(System Item) + sizeof(B_4 Picture Item)

8.4 Application of Random Index Pack

The Random Index Pack is a device to help find partitions scattered throughout an MXF file. It is a fixed length pack which defines the BodySID and Byte Offset to the start of each partition (i.e. the first byte of the partition pack key). This pack can be used by decoders to rapidly access Index Tables and to find the partitions to which an Index Table points. The Random Index Pack is optional and if it exists it follows the Footer Partition and is the last KLV item in a file.

In this specification, the Random Index Pack is strongly recommended. Because of the variable length Edit Unit, it could be much difficult to find each element without the Random Index Pack.

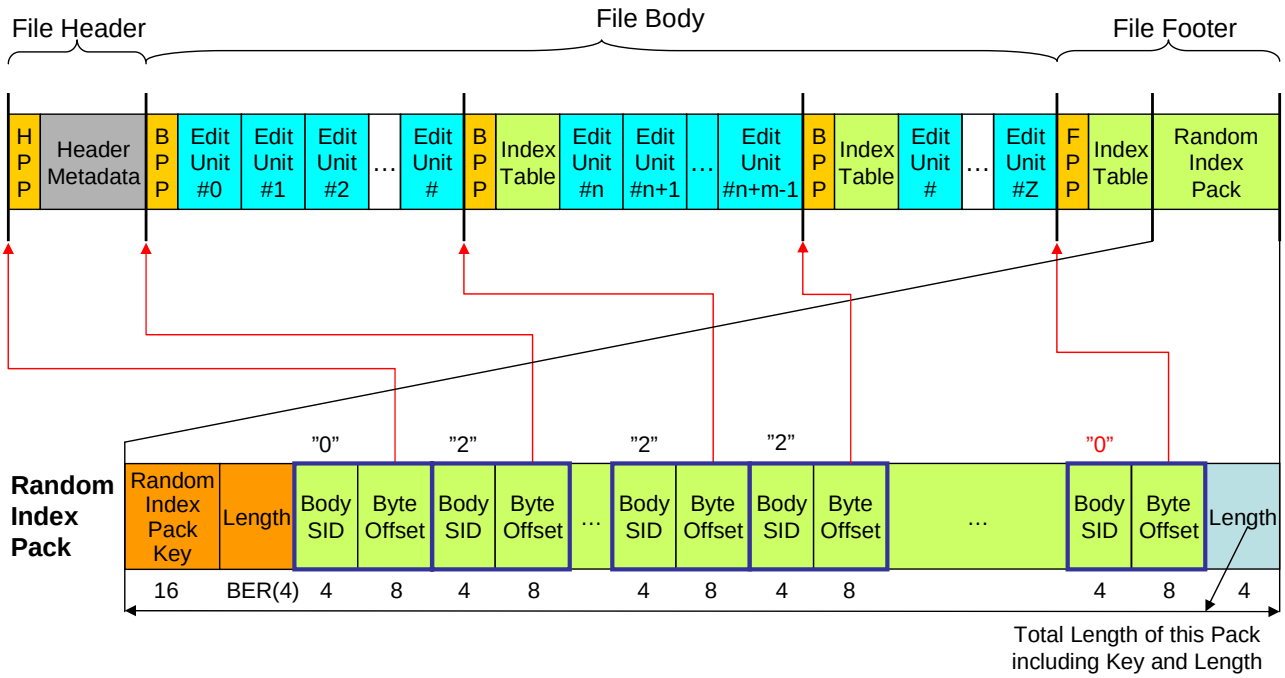


Figure 12 – Random Index Pack

Annex A
UL Code List

The following is a sample list of UL codes used for by Sony MPEG Long GOP products on for transmission.

Header Partition Pack	Closed Complete	06 0e 2b 34 02 05 01 01 0d 01 02 01 01 02 04 00
Operational Pattern	1a	06 0e 2b 34 04 01 01 01 0d 01 02 01 01 01 09 00
Essence Containers	MPEG ES Video Frame Wrapped	06 0e 2b 34 04 01 01 02 0d 01 03 01 02 04 60 01
	AES Audio Frame Wrapped	06 0e 2b 34 04 01 01 01 0d 01 03 01 02 06 03 00
Fill Item		06 0e 2b 34 01 01 01 02 03 01 02 10 01 00 00 00
Header Metadata	Primer Pack	06 0e 2b 34 02 05 01 01 0d 01 02 01 01 05 01 00
	Preface Set	06 0e 2b 34 02 53 01 01 0d 01 01 01 01 01 2f 00
	Identification Set	06 0e 2b 34 02 53 01 01 0d 01 01 01 01 01 30 00
	Content Storage Set	06 0e 2b 34 02 53 01 01 0d 01 01 01 01 01 18 00
	Essence Container Data Set	06 0e 2b 34 02 53 01 01 0d 01 01 01 01 01 23 00
	Material Package Set	06 0e 2b 34 02 53 01 01 0d 01 01 01 01 01 36 00
	File Package Set	06 0e 2b 34 02 53 01 01 0d 01 01 01 01 01 37 00
	Generic Picture Essence Descriptor Set	06 0e 2b 34 02 53 01 01 0d 01 01 01 01 01 27 00
	Picture Essence Coding (MPEG-2)	06 0e 2b 34 04 01 01 03 04 01 02 02 01 04 03 00
	Multiple Descriptor Set	06 0e 2b 34 02 53 01 01 0d 01 01 01 01 01 44 00
	Essence Container (Multiple EC UL)	06 0e 2b 34 04 01 01 03 0d 01 03 01 02 7f 01 00
	MPEG Video Descriptor Set	06 0e 2b 34 02 53 01 01 0d 01 01 01 01 01 51 00
	AES3 Audio Descriptor Set	06 0e 2b 34 02 53 01 01 0d 01 01 01 01 01 47 00
	Timecode Definition	06 0e 2b 34 04 01 01 01 01 03 02 01 01 00 00 00
	Picture Definition	06 0e 2b 34 04 01 01 01 01 03 02 02 01 00 00 00
	Sound Definition	06 0e 2b 34 04 01 01 01 01 03 02 02 02 00 00 00
	Track Set	06 0e 2b 34 02 53 01 01 0d 01 01 01 01 01 3b 00
	Sequence Set	06 0e 2b 34 02 53 01 01 0d 01 01 01 01 01 0f 00
	Source Clip Set	06 0e 2b 34 02 53 01 01 0d 01 01 01 01 01 11 00
	Timecode 12M Component Set	06 0e 2b 34 02 53 01 01 0d 01 01 01 01 01 14 00
	XML Document Text[4]	06 0e 2b 34 01 01 01 05 03 01 02 20 01 00 00 00
Body Partition Pack	Closed Complete	06 0e 2b 34 02 05 01 01 0d 01 02 01 01 03 04 00
Index Table Segment	Index Table Segment Key	06 0e 2b 34 02 53 01 01 0d 01 02 01 01 10 01 00
System Item	System Metadata Pack	06 0e 2b 34 02 05 01 01 0d 01 03 01 04 01 01 00
	Package Metadata Set	06 0e 2b 34 02 43 01 01 0d 01 03 01 04 01 02 xx
Picture Item	MPEG Frame Wrapped Picture Element	06 0e 2b 34 01 02 01 01 0d 01 03 01 15 01 05 00
Sound Item	AES Frame Wrapped Sound Element	06 0e 2b 34 01 02 01 01 0d 01 03 01 16 04 03 0x
Footer Partition Pack	Closed Complete	06 0e 2b 34 02 05 01 01 0d 01 02 01 01 04 04 00
Random Index Pack	Random Index Pack Key	06 0e 2b 34 02 05 01 01 0d 01 02 01 01 11 01 00

[4] This key is for a stand alone KLV packet whose value contains XML document text.

There is a legacy base of MXF files that use a different Fill item key. This key has the value 06.0e.2b.34.01.01.01.01.03.01.02.10.01.00.00.00. MXF decoders should be able to recognize both Fill item keys.

Annex B
Bibliography

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

SMPTE RP 210, Metadata Dictionary Registry of Metadata Element Descriptions

SMPTE RP 224, SMPTE Labels Register