EBU – Recommendation R 122

Material Exchange Format Timecode Implementation

Version 2.0

Source: EC-I SP/HIPS MXF

Geneva
November 2010

Page intentionally left blank. This document is paginated for two sided printing

Conformance Notation

This document contains both **normative** text and **informative** text.

All text is normative except for that in the Introduction, any section explicitly labelled as 'Informative' or individual paragraphs that start with 'Note:'.

Normative text describes indispensable or mandatory elements. It contains the conformance keywords 'shall', 'should' or 'may', defined as follows:

'Shall' and 'shall not':	Indicate requirements to be followed strictly and from which no deviation is permitted in order to conform to the document.
'Should' and 'should not':	Indicate that, among several possibilities, one is recommended as particularly suitable, without mentioning or excluding others. OR indicate that a certain course of action is preferred but not necessarily required. OR indicate that (in the negative form) a certain possibility or course of action is deprecated but not prohibited.
'May' and 'need not'	Indicate a course of action permissible within the limits of the document.

Default identifies mandatory (in phrases containing "shall") or recommended (in phrases containing "should") presets that can, optionally, be overwritten by user action or supplemented with other options in advanced applications. Mandatory defaults must be supported. The support of recommended defaults is preferred, but not necessarily required.

Informative text is potentially helpful to the user, but it is not indispensable and it can be removed, changed or added editorially without affecting the normative text. Informative text does not contain any conformance keywords.

A conformant implementation is one that includes all mandatory provisions ('shall') and, if implemented, all recommended provisions ('should') as described. A conformant implementation need not implement optional provisions ('may') and need not implement them as described.

Page intentionally left blank. This document is paginated for two sided printing

Contents

Page intentionally left blank. This document is paginated for two sided printing

Material Exchange Format

Timecode implementation

EBU Committee	First Issued	Revised	Re-issued
(PMC) EC-I	(2007)	2010	

Keywords: MXF, Material Exchange Format, Timecode

1. Scope

While this document contains specific provisions for the production essence formats that are recommended by the EBU in its technical documents, many of the definitions in this document are also applicable to MXF files that contain emerging essence formats, for which the EBU has yet to make recommendations.

This recommendation defines an encoding of source timecode into MXF files that is independent of any previous processing or material transfer steps. For MXF encoders, this recommendation defines the placement of source timecode in the MXF header metadata for all currently defined operational patterns. It also defines the placement of source timecode in frame-wrapped and, for EBU recommended essence formats, in clip-wrapped essence containers.

For MXF decoders and other applications that make timecode-based references into the material represented by MXF files, this document defines rules that enable consistent application behaviour with files generated by MXF encoders complying with this specification as well as with files generated by standards-compliant MXF encoders that do not follow the encoder provisions contained in this recommendation. The recommendation is not applicable where MXF decoders are required to access timecode stored in lower level source packages (e.g. from previous material transfer steps, from partial restore applications or editing applications supporting essence tracking) in order to control access to material or generate output timecode.

This recommendation provides a template to be used by manufacturers to document the compliance of the timecode implementation of their MXF encoders and decoders.

2. Introduction

The current specifications of the Material Exchange Format and their normative references [see Bibliography] provide for a number of different ways of carrying timecode information in MXF files.

Timecode can be located in the header metadata (material package timecode track and source package timecode tracks), in a Generic Container system item (e.g. based on SDTI source data) and in the payload of picture, compound, sound and data elements.

The EBU's MXF Implementation Tests have shown that current MXF implementations make use of all these different timecode carriage options [EBU BNP 071]. Given that fact, it could not be expected at the time of developing this recommendation that:

- MXF encoders will place primary timecode information (such as LTC) or other, secondary timecode information (such as VITC) consistently,
- MXF decoders will generate output timecode consistently,
- Applications that make timecode-based references into the material represented by MXF files (e.g. archive or content management systems) will use the same reference timecode location.

Harmonisation of timecode implementations is needed to obtain predictable and reliable results when interchanging MXF files and when integrating products of manufacturers into a TV production system.

The EBU has been investigating example applications and has collected user requirements. The analysis of these requirements and their translation into MXF implementation constraints has led to this recommendation, which users and implementers are encouraged to follow in their production systems and in implementing their products.

3. Recommendations

The EBU recommends the following implementation parameters in order to support consistent use of timecode in MXF applications.

Recommendations for MXF encoders

1) Timecode source

When generating MXF files, it shall be possible to select a timecode source. Depending on the product and application, the selection shall be possible from one or more of the following

a) Vertical interval timecode (VITC) according to SMPTE 12M-1-2008

b) Ancillary Time Code (ATC, formerly known as DVITC) according to SMPTE 12M-2-2008

c) Linear timecode (LTC) according to SMPTE 12M-1-2008

d) Timecode from the application controlling the MXF encoder (e.g. real-time recording device or software encoder). Examples for such interfaces are the Sony 9 pin protocol, VDCP or other appropriate application programmable interfaces (API).

e) Preset timecode

Note 1: Some products may be able to record multiple timecode sources. Apart from 2)e), the placement of secondary and ternary timecodes in MXF files and the use of such timecodes by MXF decoders are outside the scope of this document. This document's provisions are only applicable to the (primary) source timecode.

Note 2: SMPTE timecode as defined in SMPTE 12M-1 may be presented in two forms, LTC, or VITC. These two formats are strongly influenced by historic usage. LTC has it roots in the recording of time code on the longitudinal track of a VTR. LTC is commonly referred to as linear timecode. VITC (vertical interval timecode) was a means to carry the time code closely bound to the video signal. There have been various variants of VITC, one of them, DVITC, is no longer in widespread use, it was closely aligned to SDTV signals. The time information conveyed by either of these two forms is the same; however other information such as field flags, is different. Only the VITC form can be used to convey two-frame information when the time code represents progressive image formats beyond 30 frames per second.

Note 3: Carriage of either form of timecode over a serial interface involves the use of an ancillary data packet that holds timecode data, called an ATC packet. This is defined in SMPTE 12M-2. The ATC packet can convey either LTC or VITC time code. ATC itself is not a timecode rather a wrapper into which timecode and other data may be mapped. The location of the ATC packet in the serial bit stream is not defined. Any application must detect the DID of the ancillary data packet in order to extract the timecode.

2) Source timecode representation in MXF header metadata

a) When recording, the source timecode time addresses for each essence container shall be represented on a timecode track in the top level source package that describes this essence container.

b) In a source package with more than one timecode track, the source timecode track shall be identified by setting the track number property to 1. There shall be only one timecode track with a track number property value of 1 in a package. In a source package where the source timecode track is the only timecode track, the track number property may be set to 0.

c) The source timecode track should contain a frame-accurate representation of the source timecode time addresses.

d) MXF encoders should generate a timecode track for each material package. For OP1a, OP2a, OP3a and OP Atom files, the default start timecode time address of the material package timecode track should be equal to the timecode time address of the source package position that is referenced by the start of the first material package source clip.

e) Where the source timecode is provided by the controlling application [see 1)d)] or preset [see 1)e)] and when LTC, VITC or ATC are available to the MXF encoder, users should be able to select one of the additional available timecode interfaces to be encoded as historical source timecode on the source timecode track of a lower level source package that is referenced by the top level source package. Implementations that support this functionality shall also follow the provisions of items a) through d) above.

f) If an MXF encoder does not support any of the timecode sources identified in 1) or if an MXF encoder does not receive any source timecode at the selected (configured) source timecode interface, the source package timecode track may be omitted. The source package timecode track shall be present in all other cases (see a)).

3) Source timecode representation in MXF essence containers

Only in the special case where the MXF encoder both does not have an interface to receive SMPTE 12M-1 binary groups or binary group flags and where the MXF encoder does generate a frame accurate source timecode representation in the header metadata, the essence container time code may be omitted.

In all other cases, the representation of the source timecode in MXF essence containers shall be frame accurate including user bits and binary flags according to the following rules:

a) <u>Frame wrapped essence containers</u>

i) The source timecode should be encoded in the MXF Generic Container system item either as:

(1) SMPTE 385M User Date/Time property or

(2) First timecode in the SMPTE 405M timecode array of the SMPTE 394M System Element.

There shall never be both SMPTE 385M and SMPTE 405M system item elements in the same essence container.

ii) There should be no timecode within essence streams. In case the essence streams byte syntax specifies timecode, it should be set to a constant value of 00:00:00:00.

iii) Source timecode that is present within essence streams (e.g. SMPTE 328M MPEG ES Editing Information or VITC) should correspond to the source timecode that is also present in the system item.

Note 4: To update timecodes embedded in the compressed stream requires additional effort (in implementation and processing). However, if such timecode values are not updated when generating new files, surprising results may occur at later stages in the production chain. Applications should not use timecode that is embedded in essence of edited material.

iv) MXF encoders that do not encode the MXF Generic Container system item (as recommended in 3)a)i) shall encode source time code as defined in specified in Table 1.

For essence containers for which no standardised way to carry timecode exists (i.e. for which there is no entry in Table 1) source timecode shall be carried in the MXF Generic Container system item according to the rules defined in the sub-items of 3)a)i).

b) Clip wrapped essence containers

i) For the essence stream formats recommended by EBU for production use, the source timecode shall be encoded as defined in Table 1.

Table 1: Time code location in essence streams

Essence stream format	Source timecode carriage mechanism
AVC (ISO/IEC 14496-10)	Timecode can be carried in AVC SEI messages. However, for this no standardised method is available at the time of publication of this document
VC-3 (SMPTE 2019)	Time code control data in the header
DV (IEC 61834-2)	Title Timecode pack and Title Binary Group pack in the DV subcode main area according to IEC 61834 4
DV-based (SMPTE 314M and 370M)	Time code pack and binary group pack in the DV subcode section according to SMPTE 314M and SMPTE 370M
JPEG 2000 (ISO/IEC 15444-1)	No timecode can be carried inside the JPEG 2000 stream. Note: At the time of publication of this document Amendment 4 for ISO/IEC 15444-1:2004 “Profiles for Broadcast Applications” is currently developed. It is assumed that after publication of this amendment, there will be a standardised way to carry timecode inside the JPEG 2000 stream
MPEG 2 Video ES	Reference date time stamp according to SMPTE 328M
Uncompressed video	Either as VITC or ATC, but never both

Note 5: Some essence formats may not provide mechanisms to carry timecode information. For those formats, it is not possible to represent timecode information in clip wrapped essence containers.

Recommendations for MXF decoders

4) Accessing (e.g. playing) content of MXF files

Regardless of their operational pattern, all MXF files contain at least one material package and one source package. MXF files may contain multiple material and source packages.

a) If the Primary Package property of the Preface is present and if its value references an internal material package or an internal top-level file package, the referenced package shall be the default package for accessing (e.g. playing) the file – regardless of the operational pattern of the file.

Note 6: For OP Atom files, the default package for accessing (e.g. playing) shall be the top level file package. This is because SMPTE 390M defines that the Primary Package property of OP Atom files must be set to the Package ID value of the file's top level source package, and SMPTE 377-1 defines that the package referenced by the Primary Package property shall be the default package for playing the file.

Note 7: Depending on the complexity and structure of their essence container and other constraints, some generalised operational pattern files (i.e. OP1a, OP2a, OP3a, OP1b, OP2b or OP3b) files can also be valid OP Atom files except for their OP label and, possibly, the Primary Package property of the Preface. If such an OP1a, OP2a, or OP3a is relabelled as OP Atom, SMPTE 390M requires the Primary Package property of the Preface to be present and to reference the top-level file package. If the Primary Package property is added or modified as part of the relabeling to OP Atom, the SMPTE standards require a different default playing behaviour for the resulting OP Atom file (i.e. compare 4)a) and 4)b)i). The same applies when an OP Atom file is relabelled with the appropriate generalised optional pattern label and the Primary Package property is modified in the process. If the Primary package property is not changed, the default playing behaviour remains identical.

Note 8: Future revisions of SMPTE 390M-2004 may or may not result in a change of the default package for playing OP Atom files.

b) If the Primary Package property of the Preface is not present, or if it does not reference an internal material package or an internal top-level file package, the default package for accessing (e.g. playing) the file shall be determined according to the following rules:

i) For generalised operational pattern files with one material package (i.e. OP1a, OP2a, OP3a, OP1b, OP2b, OP3b), the default package for accessing (e.g. playing) shall be the material package.

ii) For generalised operational pattern files with more than one material package (i.e. OP1c, OP2c and OP3c), applications shall always access (e.g. play) a specific package of the file.

Note 9: In MXF, all packages are uniquely identified by the PackageID property. The value of this property is a 32 byte long SMPTE 330M Unique Material Identifier (UMID). Each Package KLV set is also uniquely identified by its Instance UID property. The value of this property is a 16 byte long ISO/IEC 11578:1996, Annex A Universal Unique Identifier (UUID).

c) Advanced applications may allow users or system integrators to select the package to be accessed.

5) Timecode-controlled access of content represented by MXF files

a) Applications that access (e.g. play) content based on timecode time addresses shall use the timecode track of the accessed (e.g. played) package as the default timecode to control the operation.

Note 10: Depending on the operational pattern and the value of the Primary Package property of the file (see 4)a)), this can be the timecode of a material package or the timecode of a source package.

Note 11: Material package timecode is guaranteed to be linear. Source package timecode may contain discontinuities.

i) For source packages with multiple timecode tracks, the application shall use the timecode track for which the track number property equals 1 [see 2)b)].

If none of the timecode tracks has a track number property value of 1, the application shall use the timecode track that is referenced first in the tracks property of the package (i.e. has the lowest index in the array of strong references to tracks).

ii) If the played package does not have a timecode track, the application shall simulate linear timecode. The default start value of this timecode should be 10:00:00:00. The user should be able to configure this start value.

b) For playing the material packages of OP1a, OP2a or OP3a files or for playing top level file packages that describe internal essence, advanced applications may be configured to use the essence container timecode of the played package to control the operation.

Note 12: Essence container timecode may contain discontinuities.

Note 13: In frame wrapped essence containers, the essence container timecode is the timecode stored in the system item [see 3)a)]. In clip wrapped essence containers, the essence container timecode is the timecode stored in the essence stream [see 3)b)]

Note 14: Users and implementers need to be aware that, unless explicitly forbidden by the user's application specification, top level source packages of generalised operational pattern files (i.e. operational patterns 1, 2 and 3, a, b and c) can describe essence that is external to the file.

For frame-wrapped essence containers that do not have a system item, MXF decoders shall use the timecode specified in Table 1 as the source timecode stored in the essence container.

If no timecode is found in the essence container, the control timecode shall be generated according to 5)a)ii).

6) Output timecode

a) Output timecode interfaces may be:

i) Linear timecode (LTC)

ii) Vertical interval timecode (VITC)

iii) Ancillary Time Code (ATC)

iv) Timecode emitted via an appropriate API

Applications that play MXF files shall generate output timecode according to the following rules:

b) The default output timecode shall be equal to the control timecode defined in section 5).

c) Advanced applications may optionally allow the output timecode to be configured as:

i) Preset timecode

ii) Source timecode stored in the essence container that is described by the played package (i.e. the material package of an OP1a, OP2a or OP3a file or a source package of a file of arbitrary operational pattern).

Note 15: The representation of source timecode in essence containers is defined in section 3).

Note 16: In frame wrapped essence containers, the essence container timecode is the timecode stored in the system item [see 3)a)]. In clip wrapped essence containers, the essence container timecode is the timecode stored in the essence stream [see 3)b)]

Note 17: Users and implementers need to be aware that, unless explicitly forbidden by the user's application specification, top level source packages of generalised operational pattern files (i.e. operational patterns 1, 2 and 3, a, b and c) can describe essence that is external to the file.

For frame-wrapped essence containers that do not have a system item, MXF decoders shall use the timecode specified for the clip-wrapped essence of the same essence type as the source timecode stored in the essence container [see 3)b)].

If no timecode is found in the essence container, the output timecode shall be generated according to 5)a)ii).

Operational pattern specific recommendations

7) Applications that convert interleaved OP1a files into a set of OP Atom or mono-essence OP1a files should preserve material package and source package timecode.

Note 18: If an OP1a file does not have identical timecode time addresses in the material and source packages, this may result in surprising results when playing the OP Atom file due to the different default played package (see recommendations 4.a and 4.b).

8) When converting OP Atom or mono-essence OP1a files into interleaved OP1a files, the following rules shall apply:

a) If all OP Atom or mono-essence OP1a files have the same source package timecode, the source package timecode of the OP1a file shall be identical to the source package timecode of the OP Atom or mono-essence OP1a files.

b) If all OP Atom or mono-essence OP1a files have the same material package timecode, the material package timecode of the OP1a file shall be identical to the material package timecode of the OP Atom or mono-essence OP1a files.

Application specific recommendations

9) Partial restore applications

Partial restore applications should preserve timecode such that:

a) the timecode of the material package of the new file is equal to the corresponding timecode of the material package in the original file,

b) the timecode of the source package of the new file is equal to the corresponding timecode of the source package in the original file and,

c) the timecode in the essence container of the new file is equal to the essence container timecode of the corresponding material in the original file.

10) Progressive systems with frame rates greater than 30 frames per second[1]

For progressive systems with frame rates greater than 30 frames per second, MXF encoders and MXF decoders shall apply SMPTE 12-1-2008 such that:

a) timecode inside MXF Generic Container system item (either SMPTE 385M User Date/Time property or first timecode in the SMPTE 405M timecode array of the SMPTE 394M System Element)

b) timecode within essence stream that do not support frame rates greater than 30 frames per second but do support a field mark flag

are compliant with the following rules:

[1] Please note that the exact value is 39 frames per second. However, for consistency reasons with SMPTE 12M-1-2008 the value 30 frames per second is used.

the timecode should increment only every other frame. In this case the field mark flag shall be used to identify each frame of the frame pair. The field mark flag for the first frame of the frame pair shall be set to zero, the field mark flag for the second frame of the frame pair shall be set to one. Documentation recommendation for vendors

11) As part of their product documentation, vendors should provide detailed documentation about the parameters and functionalities of the timecode implementation [see Annex E].

4. Bibliography

SMPTE 12M-1-2008	Television — Time and Control Code
SMPTE 12M-2-2008	Television — Ancillary Data Mapping over MPEG-2 Video Elementary Stream Editing Information
SMPTE 314M-2005	Television — Data Structure for DV-Based Audio, Data and Compressed Video — 25 and 50 Mbit/s
SMPTE 328M 2000	Television — MPEG-2 Video Elementary Stream Editing Information
SMPTE 330M-2004	Television — Unique Material Identifier (UMID)
SMPTE 370M-2008	Television — Data Structure for DV-Based Audio, Data and Compressed Video at 100 Mbit/s 1080/60i, 1080/50i, 720/60p, 720/50p
SMPTE 377-1-2009	Television — Material Exchange Format (MXF) — File Format Specification (Standard)
SMPTE 379-1-2009	Television — Material Exchange Format (MXF) — MXF Generic Container
SMPTE 379-2-2010	Television — Material Exchange Format (MXF) — MXF Constrained Generic Container
SMPTE 380M 2004	Television — Material Exchange Format (MXF) – Descriptive Metadata Scheme-1 (Standard, Dynamic)
SMPTE 385M 2004	Television — Material Exchange Format (MXF) – Mapping SDTI-CP Essence and Metadata into the MXF Generic Container
SMPTE 390M 2004	Television — Material Exchange Format (MXF) – Specialized Operational Pattern “Atom” (Simplified Representation of a Single Item)
SMPTE 394M 2006	Television — Material Exchange Format (MXF) – System Scheme 1 for the MXF Generic Container
SMPTE 405M 2006	Television — Material Exchange Format (MXF) – Elements and Individual Data Items for the MXF Generic Container System Scheme 1
SMPTE 2019-4	Television — Material Exchange Format (MXF) — Mapping VC-3 Coding Units into the MXF Generic Container
SMPTE 2031	Television — Material Exchange Format (MXF) — Mapping VC-1 into the MXF Generic Container
RP 2008-2008	Television — Material Exchange Format (MXF) — Mapping AVC Streams into the MXF Generic Container
ISO/IEC 11578:1996, Annex A	Information technology — Open System Interconnection — Remote Procedure Call (RPC), Universal Unique Identifier (UUID)
EBU BPN 071	EBU MXF Implementation tests, May 2006 (EBU Members only)

Annex A: User requirements for the application of timecode in MXF

The EBU has identified the following user requirements for the carriage of timecode in MXF files.

These requirements take into account interoperability in heterogeneous systems that mix file based operations and stream/tape based operations. In addition, they consider applications that employ timecode as reference to manage and annotate material contained in MXF files.

1) It is required to have consistent behaviour of timecode at each device (MXF encoders, MXF decoders, applications manipulating MXF files and applications referencing into MXF files) in the production chain.

2) If possible, a single solution should be recommended for the transport of timecode in MXF files.

3) It has to be possible to select the leading timecode source. Depending on the product and application, this can be vertical interval timecode (VITC), digital vertical interval timecode (ATC) according to SMPTE 12M-2-2008, linear timecode (LTC), timecode from the application controlling the recording device (e.g. via Sony 9 pin protocol or VDCP) or preset timecode.

4) Regardless of its source, the leading timecode time addresses need to be recorded transparently, i.e. including a frame accurate representation of timecode discontinuities.

5) For applications that rely on SMPTE 12M-1 timecode to carry application information other than timecode time addresses, the user-bits and binary group flags need to be recorded and played transparently.

6) The functionality to use preset timecode should be supported by all applications.

7) Timecode-based control behaviour (e.g. for play, partial copy, etc.) should be consistent regardless of which system the MXF file was generated.

8) LTC should be supported for play-out applications (control of the MXF, set start values etc.)

9) If no single solution can be recommended, users should be able to select the leading timecode for the output interface, especially if MXF files contain different timecodes in different locations.

10) If no single solution can be recommended, users should be able to select the mechanism where the leading timecode is stored within the MXF file.

11) It is required to have frame accurate timecode support also for progressive systems with frame rates greater than 30 frames per second.

12) Vendors must provide information about the timecode handling by their MXF encoders (e.g. the timecode sources and timecode locations within the MXF file) and MXF decoders (e.g. file timecode locations used to generate output timecode). This includes information about the support of frame accurate timecode representation, user bits and binary group flags.

Page intentionally left blank. This document is paginated for two sided printing

Annex B: MXF Timecode Carriage Mechanisms

The following figure contains an illustration of the different ways, in which Timecode can be represented in MXF files.

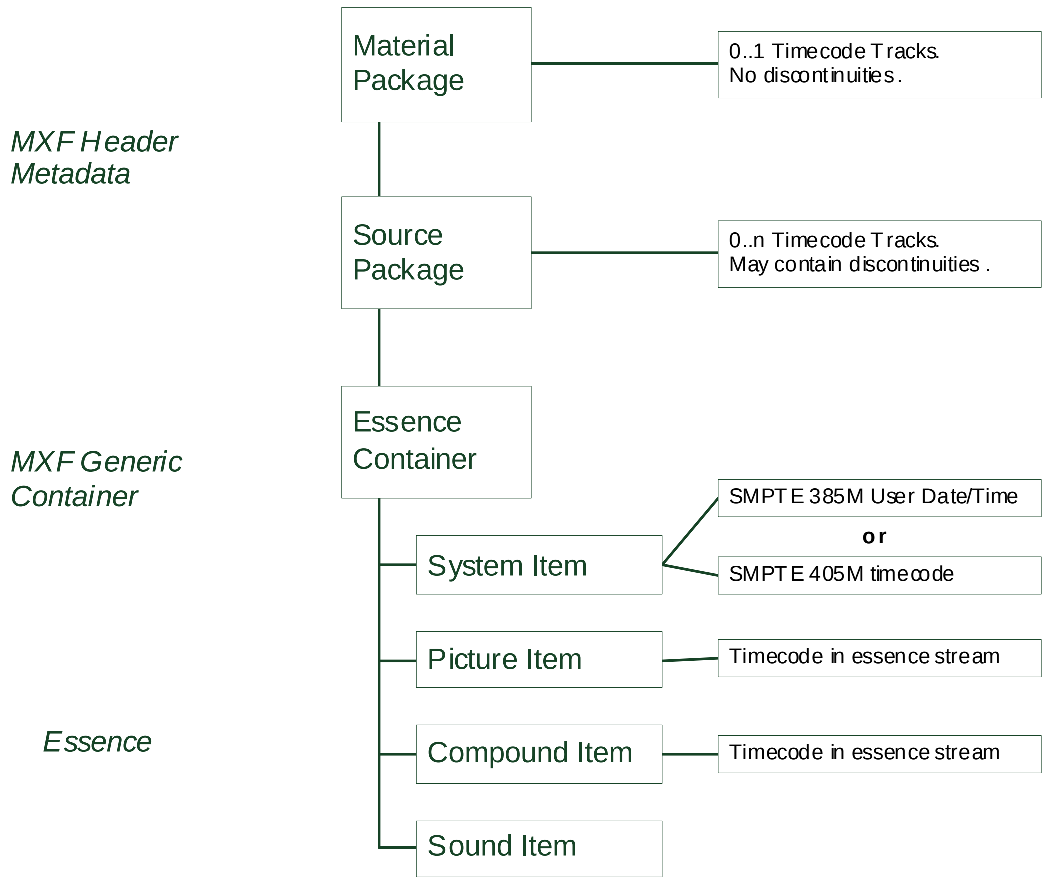

Figure 1: MXF Timecode Carriage Mechanisms

MXF header metadata can have zero or one timecode track per material package and zero to many timecode tracks per source package. While material package timecode tracks can only represent continuous timecode, source package timecode tracks can represent zero to many timecode discontinuities. Both material package and source package timecode tracks carry “synthetic timecode information” (i.e. they only represent timecode time addresses, but no SMPTE 12M-1 user bits or binary group flags).

The system item of the MXF Generic Container can carry two timecode streams (i.e. User Date/Time and Creation Date/Time). These streams consist of one timecode value per content package. They are able to carry all bits of the SMPTE 12M-1 timecode words and it is possible to signal with which specifications these streams comply (i.e. SMPTE 12M-1 or SMPTE 309M). If frame wrapping is used, the system item timecodes are able to frame accurately carry timecode, regardless of the number of discontinuities.

Many of the payloads that can be carried inside Generic Container elements provide additional timecode carriage mechanisms. The following list contains the most prominent examples:

- MPEG Video ES streams can carry one timecode time address in each GOP header and two SMPTE 12M-1 timecode words via SMPTE 328M MPEG Video ES Editing Information in the MPEG ES picture user data.

- DV-DIF streams can carry SMPTE 12M-1 timecode information (including binary groups and user bits) in the Subcode and VAUX area.
- Picture formats that encode part of the vertical blanking interval (VBI) can carry VITC timecode. For 625 line SDI signals, D10-type streams encode VBI lines 7 through 22 and 330 through 335; DV-DIF streams always carry line 335; DV-DIF streams can also carry one to 4 VBI lines according to SMPTE 374M (depending on bitrate on the compressed stream and line format).
- VITC or ATC can also be carried inside the payload of Generic Container data elements.

Annex C: Current Implementations

In the items below, a number of example applications are analysed with respect to their current implementation of timecode.

1) Recording of live streams

In current implementations, the following timecode implementations have been observed as result of the EBU MXF Implementation Tests:

a) LTC represented in Generic Container system item (User Date/Time)

b) LTC represented in payload of compressed picture element (D10 with SMPTE 328M MPEG Video ES Editing Information or DV-DIF Subcode Packs)

c) VITC in payload of compressed picture element (VBI lines of D10 MPEG ES or DV-DIF VAUX Packs or line 335 of compressed DV-DIF stream)

d) Continuous timecode (based on LTC start timecode or preset to values such as 10:00:00:00) in material package timecode track

e) Continuous timecode (based on LTC start timecode or preset to values such as 10:00:00:00) source package timecode track. Timecode discontinuities in the leading timecode were not represented.

2) Play-out of MXF files as a live stream

In current implementations, the following timecode implementations have been observed as result of the EBU MXF Implementation Test:

a) Output timecode (LTC) is generated from the source package timecode track.

b) Output timecode (LTC) is generated from the Generic Container system item User Date/Time.

c) Output timecode (LTC) is generated with no relationship to any of the timecodes in the MXF file.

d) Output timecode (VITC) that is present in the compressed bit stream is regenerated at the SDI output.

e) Output timecode (VITC) that is present in the compressed bit stream is not regenerated at the SDI output.

Page intentionally left blank. This document is paginated for two sided printing

Annex D: Definition of Terms

Term	Definition
Leading timecode	The reference timecode for timecode-based operations.
Timecode time addresses	SMPTE 12M-1 timecode consists of 64 data bits (80 bits including 16 sync word bits for LTC and 90 bits including 18 sync bits and 8 CRC bits for VITC). 26 bits specify hours, minutes, seconds and frames. The combination of these values is the time address of the SMPTE 12M-1 timecode value.
Timecode binary flags	SMPTE 12M-1 timecode consists of 64 data bits. 3 bits are the so-called binary flags.
Timecode binary groups	SMPTE 12M-1 timecode consists of 64 data bits. 32 bits constitute called binary groups 1 through 8. The binary groups can be used to carry auxiliary information or application information.
Frame accurate timecode representation	The representation of a timecode source is frame accurate if all timecode time addresses and, if available, all timecode binary flags and binary group bits are a precise, frame-by-frame representation of the timecode source. For encoders this means that, in the MXF file, the time code associated with each frame of video is a precise, frame-by-frame representation of the timecode to video picture association at the input of the encoder at the time of recording. For decoders this means that the output timecode is a precise frame-by-frame representation of the timecode to video picture association defined in the MXF file.

Page intentionally left blank. This document is paginated for two sided printing

Annex E: Documentation Template for vendors

TC = timecode EC = Essence Container MP = Material Package SP = Source Package

ENCODER

1. TC source [see 1) of this document]	
Can VITC be selected as the TC source?	
Can ATC be selected as the TC source?	
Can LTC be selected as the TC source?	
Can the controlling TC be selected as the TC source? If yes, what is/are the control interface(s)? (specify)	
Can preset TC be selected as the TC source? If yes, how is the start value of the preset TC configured/controlled?	
Does the encoder support multiple simultaneous TC sources? If yes, how is the primary TC source identified? (specify)	

2. Source TC in MXF header metadata [see 2) of this document]	
Does the encoder write the source TC to a TC track in the top-level SP?	
How many top level SP TC tracks does the encoder generate?	
Which unique track number does the encoder create to identify the source TC track? (specify)	
How does the encoder react to source TC discontinuities? Is it possible to frame-accurately represent them in SP TC track? If yes, are there any constraints? (specify)	
Does the encoder generate MP TC? If yes, to which value is the start TC of the MP TC track set? How can the TC track start value be controlled? (specify)	
Does the encoder support user selection of a secondary TC as historical source TC? If yes, how is the selection performed? How is the historical source TC represented in the header metadata of the MXF file? (specify)	
If the encoder supports more than one TC source, how are the values of the additional TC sources represented in the MXF file? (specify)	
What happens if the encoder does not receive any TC at the selected source TC interface? What happens if the encoder does not have a designated source TC interface?	

3. Source TC in MXF essence containers [see 3) of this document]	
Is the EC frame- or clip-wrapped? (specify)	
Is the source TC recorded in the MXF EC?	
For frame-wrapped ECs, is the source TC encoded in the generic container system item? If yes, according to which specification(s) and at which location? (specify)	
Is the source TC encoded in the essence stream? If yes, according to which specification and at which location? (specify)	
Does the TC representation in the EC include source TC user bits & binary group flags? (specify)	
How do does the encoder react to source TC discontinuities? Is it possible to frame-accurately represent them in the EC? If yes, are there any constraints? (specify)	

DECODER

4. Accessing content of MXF files [see 4) of this document]	
Does the MXF decoder use the Primary Package property of the Preface to determine the default package for accessing the file?	
In the absence of the Primary Package property (or in the absence of support of the Primary Package property), what is the default package for accessing OP1a, OP2a, OP3a, OP1b, OP2b and OP3b files? (specify)	
What is the default package for accessing OP Atom files?	
Does the decoder allow selection of a specific package for accessing contents of the MXF file? If yes, how? (specify)	

5. TC-controlled access of MXF content [see 5) of this document]	
Does the decoder use the TC from the TC track of the accessed package to control the access to the material?	
In an SP with multiple TC tracks, which track does the decoder use? (specify)	
What TC is used if there is no TC track in the package? Which options does the user have to configure the behaviour? (specify)	
When accessing the MP of OP1a, OP2a or OP3a files or when accessing an SP, is it possible to configure the decoder to use the TC in the associated EC to control the operation? If yes, how is the configuration set? (specify)	
When the application has been configured to use the TC of the ES associated with the accessed package to control the operation, what is the location of the TC in the EC(s) supported by the application? (specify)	
When the application has been configured to use the TC of the ES associated with the accessed package to control the operation, how does it react to files that do not have EC TC in the expected location? (specify)	
How does the application react to discontinuities in the TC used to control the operation? (specify)	

6. Output TC [see 6) of this document]	
Which TC interface(s) does the decoder support to output TC? (specify)	
Can the decoder use the control TC values as output TC?	
Can the decoder be configured to use preset TC as output TC?	
When playing OP1a, OP2a, OP3a MPs or SPs of arbitrary OP files, can the source TC stored in the associated EC be used to generate the output TC? If yes, how is the configuration set?	
When the application has been configured to use the TC in the associated EC as output TC, what is the location of the TC in the EC(s) supported by the application? (specify)	
When the application has been configured to use the TC in the associated EC as output TC, how does it react to files that do not have EC TC in the expected location? (specify)	

OPERATIONAL PATTERN CONVERSION

7. OP1a -> OP Atom conversion [see 7) of this document]	
Is the MP TC preserved? If not, how is the MP TC of the output files determined? (specify)	
Is the SP TC preserved? If not, how is the SP TC of the output files determined? (specify)	

8. OP Atom/mono-essence OP1a -> interleaved OP1a conversion [see 8) of this document]	
How is the MP TC of the output file determined? (specify)	
How is the SP TC of the output file determined? (specify)	

PARTIAL RESTORE

9. Application-specific [see 9) of this document]	
Is the TC of the MP of partially restored content equal to that in the source file?	
Is the TC of the SP of partially restored content equal to that in the source file?	
Is the TC of the EC of partially restored content equal to that in the source file?	