RECOMMANDATION UIT-R BT.1578

Définition du format de contenu, des éléments et des métadonnées pour les applications de production de télévision faisant intervenir des interfaces fondées sur la Recommandation UIT-R BT.1381

(Question UIT-R 5/6)

(2002)

L'Assemblée des radiocommunications de l'UIT,

considérant

a) qu'il existe déjà une Recommandation de l'UIT-R relative à une interface de transport basée sur l'interface série numérique pour les signaux de télévision comprimés en production de télévision selon les Recommandations UIT-R BT.656 et UIT-R BT.1302;

b) que les données de programme composées d'éléments audio, vidéo comprimée, données et métadonnées, sont acheminées par cette interface;

c) qu'il est nécessaire de définir des formats décrivant la structure d'assemblage des éléments système, image, audio et données auxiliaires;

d) qu'il est nécessaire de définir des possibilités d'extension de format permettant des transferts à des débits plus élevés ou moins élevés en mode de transfert isochrone ou asynchrone;

e) qu'il est nécessaire de définir un mode de synchronisation permettant de réduire les temps de propagation et d'acheminer dans chaque trame de transport deux éléments de contenu;

f) qu'il est nécessaire de prévoir la possibilité d'acheminer des éléments de contenu en mode à faible retard;

g) que des éléments de contenu de sources différentes peuvent être multiplexés en un seul flux de données;

h) qu'il devrait exister une méthode préférée de formatage des éléments de programme et des métadonnées en production de télévision,

recommande

1 lorsque des flux composés d'éléments audio, vidéo comprimée, données et métadonnées sont transportés sur des interfaces conformes à la Recommandation UIT-R BT.1381, de formater systématiquement le contenu conformément à la norme SMPTE 326M-2000 «SDTI Content Package Format, SDTI-CP»;

2 pour les applications de production et de postproduction de télévision professionnelle, de définir les éléments et les métadonnées transportés sur des interfaces conformes à la Recommandation UIT-R BT.1381 selon la norme SMPTE 331M-2000 «Element and Metadata definitions for SDTI-CP».

Note du Secrétariat: Les normes SMPTE 326M-2000 et 331M-2000, dont la version électronique était disponible sur un site web, ont été annexées au texte de cette Recommandation.

Références normatives:

La norme SMPTE 331M comporte une référence à la norme SMPTE 328M «MPEG-2 Video Elementary Stream Editing Information», laquelle à son tour fait référence à la norme SMPTE 296M «1280 × 720 Progressive Image Sample Structure – Analog and Digital Representation and Analog Interface». Les formats suivants, énumérés au Tableau 1 de la norme SMPTE 296M, ne seront pas considérés comme faisant partie de la présente Recommandation.

Elément du Tableau 1	Nomenclature du système	Fréquence d'image
3	1280 × 720/50	50
6	1280 × 720/25	25
7	1280 × 720/24	24
8	1280 × 720/23,98	24/1,001

Résumé de la norme SMPTE 326M-2000

Cette norme spécifie le format de transport des éléments de contenu (CP, *content packages*) au niveau de l'interface de transport numérique en série (SDTI, *serial digital transport interface*) d'où sa dénomination abrégée SDTI-CP. Ce format est une structure d'assemblage des divers éléments système, image, audio et données auxiliaires:

– Un élément système est un groupe d'éléments de synchronisation et de commande, auquel s'ajoutent éventuellement les métadonnées associées à l'image, à l'audio et aux données auxiliaires.

– Un élément image est un groupe pouvant comporter jusqu'à 255 éléments de flux image.

– Un élément audio est un groupe pouvant comporter jusqu'à 255 éléments de flux audio.

– Un élément données auxiliaires est un groupe pouvant comporter jusqu'à 255 éléments de données auxiliaires: lignes de données auxiliaires, télétexte, etc.

Cette norme définit la structuration des éléments de programme pour le transport SDTI. Tous les formats d'élément et de métadonnées sont définis par la norme SMPTE 331M. Un récepteur SDTI-CP conforme doit pouvoir recevoir et répartir des éléments structurés selon le format SDTI-CP.

Un décodeur SDTI-CP est défini par la capacité à recevoir et à décoder un ensemble défini d'éléments et de métadonnées d'après le document de gabarit de décodage associé à cet ensemble.

Le niveau d'application de base de cette norme est définie par le transport d'éléments de contenu, verrouillés sur la trame de transport SDTI. Cette norme définit en outre les capacités d'extension de format suivantes:

– possibilité de transfert d'éléments de contenu à des débits supérieurs ou inférieurs au débit spécifié par transfert isochrone ou asynchrone;

– disponibilité d'un mode de synchronisation permettant de réduire les temps de propagation et la possibilité de transporter deux éléments de contenu dans chaque trame de transport SDTI;

– possibilité d'acheminer les éléments de contenu en mode à faible retard; et

– multiplexage de contenus de sources différentes sur un seul élément de transport SDTI.

NOTE 1 – La norme SMPTE 326M-2000 est donnée dans l'Annexe 1 (voir aussi la Note 3).

Résumé de la norme 331M-2000

Cette norme définit les formats des éléments et des métadonnées utilisés dans la norme qui spécifie le format d'assemblage des éléments de contenu SDTI (SDTI-CP), c'est-à-dire la norme SMPTE 326M-2000, pour les éléments système, image, audio et données auxiliaires. On associe à chacun des types d'élément et de métadonnées énumérés ci-après une valeur de type d'élément ou de métadonnées. Cette valeur de type n'est pas liée aux valeurs de type de données spécifiées dans la norme SMPTE 305M. Le type de données SDTI définit un type d'élément, alors que la norme en question spécifie des types d'élément et de métadonnées.

Les fourchettes de valeurs de type d'élément et de type de métadonnées sont les suivantes:

– Elément image: fourchette des types d'élément = 01_h à $0F_h$ inclus.

– Elément audio: fourchette des types d'élément = 10_h à $1F_h$ inclus.

– Elément données auxiliaires: fourchette des types d'élément = 20_h à 77_h inclus.

– Elément système: fourchette des types d'élément = 78_h à $7F_h$ inclus.

– Elément système: fourchette des types de métadonnées = 80_h à FF_h inclus.

Une valeur de type = 00_h n'est pas une valeur élément ou métadonnées valide.

NOTE 2 – La norme SMPTE 331M-2000 est donnée dans l'Annexe 2 (voir aussi la Note 3).

NOTE 3 – Les normes SMPTE 326M-2000 et SMPTE 331M-2000 et leurs résumés renvoient uniquement aux versions 2000, qui sont celles qui ont été approuvées, le 28-06-02 en application de la Résolution UIT-R 45, par les Administrations des Etats Membres de l'UIT et par les Membres du Secteur des radiocommunications participant aux travaux de la Commission d'études 6 des radiocommunications. Comme convenu entre l'UIT et la SMPTE, lesdites versions ont été fournies par la SMPTE qui en a autorisé l'utilisation et l'UIT-R a accepté de les inclure dans la présente Recommandation. Toute version ultérieure des normes SMPTE 326M et SMPTE 331M, qui n'aurait pas été agréée et approuvée par la Commission d'études 6, n'entre pas dans le cadre de la présente Recommandation. Pour des versions ultérieures de documents SMPTE, le lecteur est prié de consulter le site web de la SMPTE à l'adresse suivante: http://www.smpte.org/.

SMPTE 326M-2000

SMPTE STANDARD

for Television — SDTI Content Package Format (SDTI-CP)

1 Scope

This standard specifies the format for the transport of content packages (CP) on the serial digital transport interface (SDTI). This format is abbreviated to the term SDTI-CP. The format is a packaging structure for the assembly of system, picture, audio, and auxiliary data items as follows:

– A system item assembled as a group of timing and control elements, plus any metadata associated with the picture, audio, and auxiliary data items;

– A picture item assembled as a group of up to 255 picture stream elements;

– An audio item assembled as a group of up to 255 audio stream elements;

– An auxiliary item assembled as a group of up to 255 auxiliary data elements such as ancillary data lines, teletext, and other data.

This standard defines the structure of the content package mapped onto the SDTI transport. All element and metadata formats are defined by SMPTE 331M.

An SDTI-CP compliant receiver shall be capable of receiving and parsing the structure of the SDTI-CP format.

An SDTI-CP compliant decoder is defined by the ability to both receive and decode a defined set of elements and metadata according to an associated decoder template document.

The baseline operation of this standard is defined by the transport of content packages locked to the SDTI transport frame rate. This standard additionally defines format extension capabilities as follows:

– Allow content package transfers at higher and lower than the specified rate through isochronous and asynchronous transfer modes;

– Provision of a timing mode to reduce delay and provision for two content packages in each SDTI transport frame;

– Carriage of content packages in a low-latency mode; and

– Multiplexing of content packages from different sources onto one SDTI transport.

This standard is limited to SDTI operating at a bit rate of 270 Mb/s and 360 Mb/s as defined by SMPTE 305M.

2 Normative references

The following standards contain provisions which, through reference in this text, constitute provisions of this standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this standard are encouraged to investigate the possibility of applying the most recent edition of the standards indicated below.

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

SMPTE 12M-1999, Television, Audio and Film — Time and Control Code

Approved
January 31, 2000

SMPTE 305M-1998, Television — Serial Data Transport Interface

SMPTE 331M-2000, Television — Element and Metadata Definitions for the SDTI-CP

SMPTE RP 168-1993, Definition of Vertical Interval Switching Point for Synchronous Video Switching

SMPTE RP 204-2000, SDTI-CP MPEG Decoder Templates

3 General specification

Figure 1 shows the basic layered structure of a content package. It shall be constructed of up to four items where each item is constructed of one or more elements.

The system item carries content package metadata and may contain a control element. The system item also carries metadata which is related to elements in the other items.

The picture item can consist of up to 255 picture stream elements.

The audio item can consist of up to 255 audio stream elements.

The auxiliary item can consist of up to 255 auxiliary data elements.

A content package contains the associated contents of one content package frame period starting with a system item and optionally containing picture, audio, and auxiliary items.

Element and metadata formats are defined in SMPTE 331M. New element and metadata types may be added as new requirements are defined.

An SDTI-CP decoder shall be specified by its ability to receive and decode an element or set of elements together with associated metadata defined by an associated decoder template document. The MPEG decoder template is SMPTE RP 204. Other decoder template recommended practices may be defined as required for other applications of the SDTI-CP. An SDTI-CP encoder which creates a content package with greater capabilities than those defined by a decoder template may risk undocumented decoder effects.

The term baseline operation is used throughout this standard and has a specific meaning defined as the combination of synchronous transfer mode operating in normal timing mode. Transfer and timing modes are defined in clauses 8 and 9.

This standard first describes baseline operation. Later parts of this standard define the methods by which the SDTI-CP format may:

– allow content package transfers at higher and lower than the specified rate through isochronous and asynchronous transfer modes;

– provide a timing mode to reduce delay and provide for two content packages in each SDTI transport frame;

– carry content packages in a low-latency mode; and

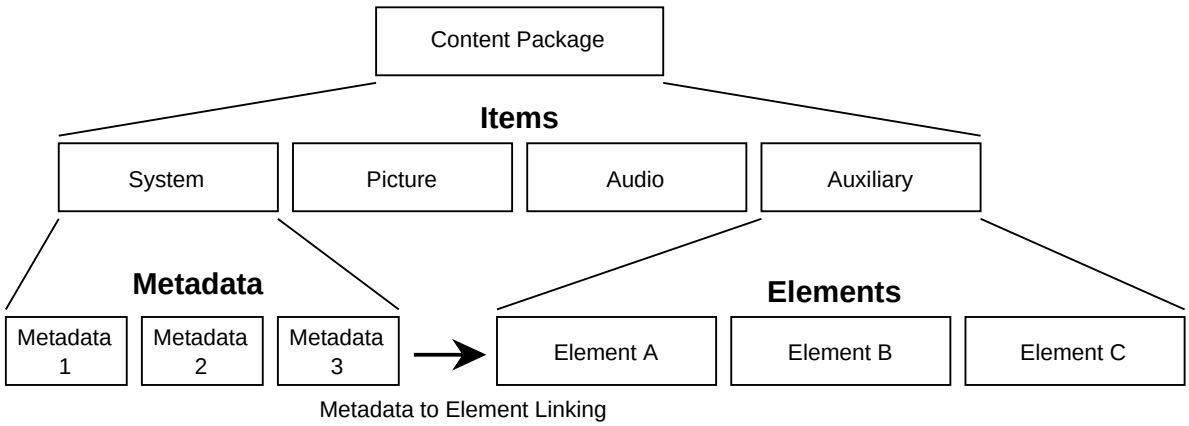

Figure 1 – Basic content package structure

– multiplex content packages from different sources onto one SDTI transport.

In general, special decoders are required to support these modes and the level of support provided shall be indicated in an associated decoder template document.

4 SDTI block structure

The system, picture, audio, and auxiliary items are each formatted as SDTI variable blocks beginning with the separator word and terminating with the end code word. Each variable block shall start immediately following the SAV sequence. Thus, the starting point of a content package can be found through the detection of a system item type code.

The format of each variable length block is shown in figure 2. The input format is 8-bit data entered into bits b0 to b7 of the 10-bit word.

Bits b8 and b9 are both set to 1 for the separator and end code words. For the item type, word count, and data block words, bit 8 is set to be the even parity of bits b0 to b7 and bit 9 is set to be the odd parity of bits b0 to b7. These specifications follow those of SMPTE 305M.

The SDTI data type word values are as follows:

– System item: 04_h
– Picture item: 05_h
– Audio item: 06_h
– Auxiliary item: 07_h

These data type values shall be registered in SMPTE 305M.

4.1 SDTI line and address numbers

Since the data in each SDTI variable block continue through as many lines as necessary until the block end, it is necessary that the SDTI header line numbers are contiguous. It is also necessary that the SDTI header source and destination address values are constant throughout the transmission of all lines associated with any one content package.

4.2 SDTI switching

The arbitrary switching of SDTI data streams, although at the picture frame boundary, may affect the ability to successfully decode picture, audio, and auxiliary data without the use of special processing equipment to mitigate the switching effects. The lines affected by a picture switch are defined in SMPTE RP 168. A continuity count is provided in the system item which can be used to indicate content packages affected by a switch.

5 Content package structure

Each content package shall consist of a minimum of the system item together with any, all, or none of the picture, audio, and auxiliary items.

The system item shall appear first in any sequence of items within a content package. The order of the picture, audio, and auxiliary items may be restricted in an associated decoder template document. There shall be only one item of any type in any one content package.

A system item shall be present in the content package. The presence of the other items in the content package is optional depending upon the transmission requirements.

In baseline operation, the content package start reference is defined by the switching point of SMPTE RP 168. The system data marks the first item of the content package and, for baseline operation, starts on the following reserved line for current television systems:

525/60: Line 13 625/50: Line 9

Figure 2 – Format of the SDTI variable block

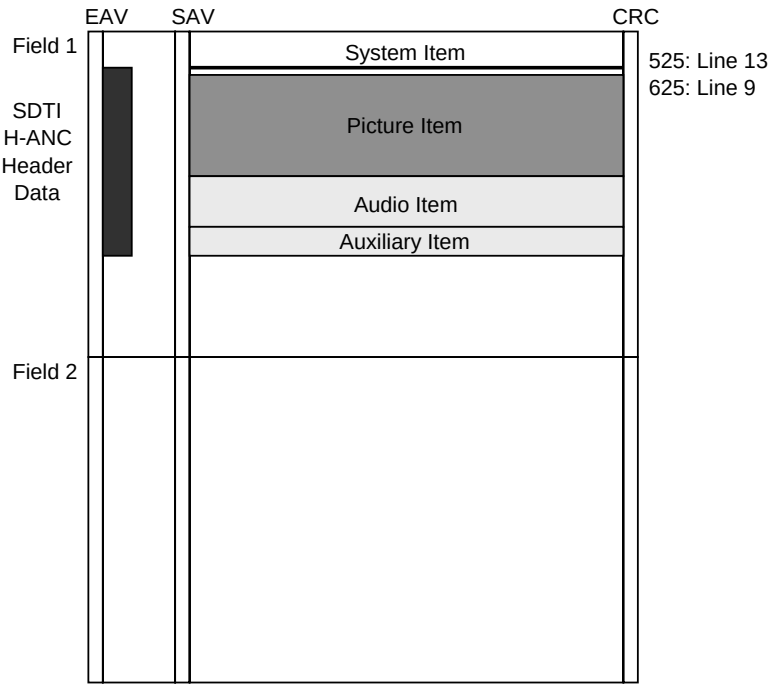

Figure 3 – Arrangement of system, picture, audio and auxiliary items in the content package

The general arrangement of system, picture, audio, and auxiliary data is given in figure 3.

The system item is provided to give content package metadata and control information together with metadata of the associated picture, audio, and auxiliary items.

A picture item is an assembly of up to 255 picture stream elements.

An audio stream item is an assembly of up to 255 audio stream elements.

An auxiliary data item is an assembly of up to 255 auxiliary data elements.

Assignment of an element to an item type is defined in SMPTE 331M. An element shall be placed in its defined item type.

The content package formed by the system, picture, audio, and auxiliary items shall represent the associated contents of one content package frame period defined by the content package rate variable in the system item.

The content package shall not exceed the frame period of the SDTI except where specifically provided by the special transfer modes described later in this standard.

The content package distribution for an example 12-frame MPEG-2 GOP is shown in figure 4.

In the case of special transfer modes, a content package may overlap the interface frame period. In some optional modes, there may be more than one content package per interface frame period. Details of these modes are available in later clauses of this standard.

5.1 Timing considerations

Each content package shall be regarded as a potentially editable unit, so time alignment of the items and elements within a content package is recommended. There may be occasions where time alignment is impractical, but it should be noted that the specification of timing misalignment between items, or elements within an item, is not automatically supported.

The transmission order for all elements in a content package is the order of transmission at the output of each respective element encoder. This may be different for each element (notably in the case of MPEG-2

Figure 4 – Content package distribution over an example MPEG-2 GOP

encoding using B frames). Users of this standard should be aware that the elements of a content package may not be time aligned on a frame-by-frame basis and should take appropriate action to avoid timing errors.

Metadata for any element within a content package shall maintain accurate frame association with the element data.

6 Picture, audio, and auxiliary item structure

The picture, audio, or auxiliary item type value is followed by a 4-byte word count and a 1-byte item header word which defines the number of elements in the item.

The item word count value for each item may be set to the length of the data block as defined by SMPTE 305M. However, when not known in advance, the word count value may be set to zero to indicate an undefined block length as defined by SMPTE 305M.

The value of the item header word is the element count and has the range 1 to 255 as shown in figure 5. An element count value of 0 is not a valid value and shall not be used. The maximum number of elements will be typically restricted by an associated decoder template document.

The element data block structure is shown in figure 6.

Each element data block starts with a 1-byte element type value, followed by a 4-byte word count of the element data block size, a 1-byte element number, and the element data.

The element word count has the same format as specified in SMPTE 305M and its value shall be the length of the element number and element data words. However, where the element word count value is not known at the point of transmission, it may be set to zero to indicate an undefined data block length.

The value of the element number shall lie in the range 0 to 255 and shall be unique among the elements within any item. It is recommended that the element number increment by one for each element in sequence in the item.

The element type and element data formats shall be as defined in SMPTE 331M.

7 System item structure

The system item contains content package metadata and control data together with metadata for each element of the picture, audio, and auxiliary items.

The system item structure is defined in figure 7. It shall start with a 7-byte system item header and may be followed by an SMPTE universal label, timing, package, picture, audio, and auxiliary metadata sets, and control as indicated by the system item bitmap word.

The seven bytes of the system item header define:

– a system item bitmap word;

– a content package rate word;

– a content package type word, including stream status flags;

– a channel handle word;

– a continuity count word.

SMPTE 326M-2000

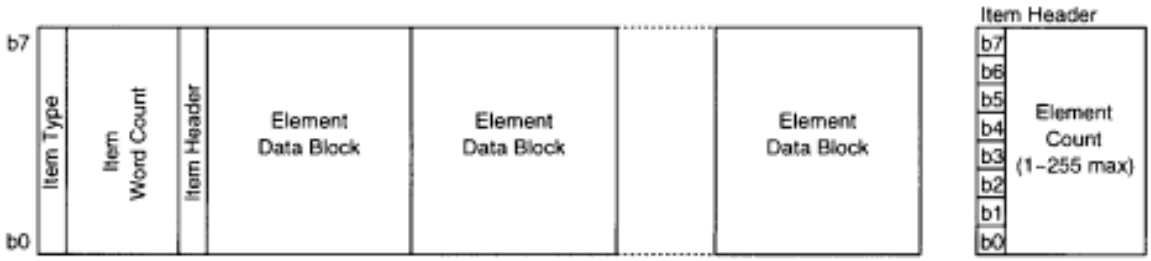

Figure 5 – Structure of picture, audio and auxiliary items

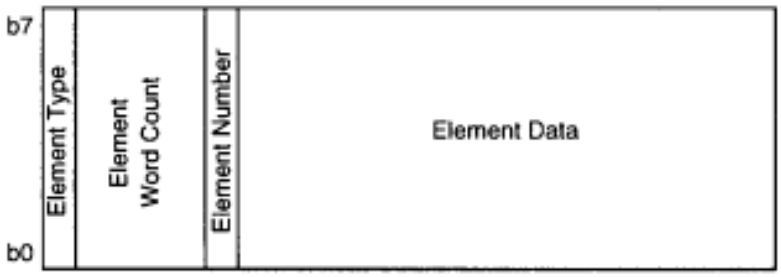

Figure 6 – Structure of element data blocks

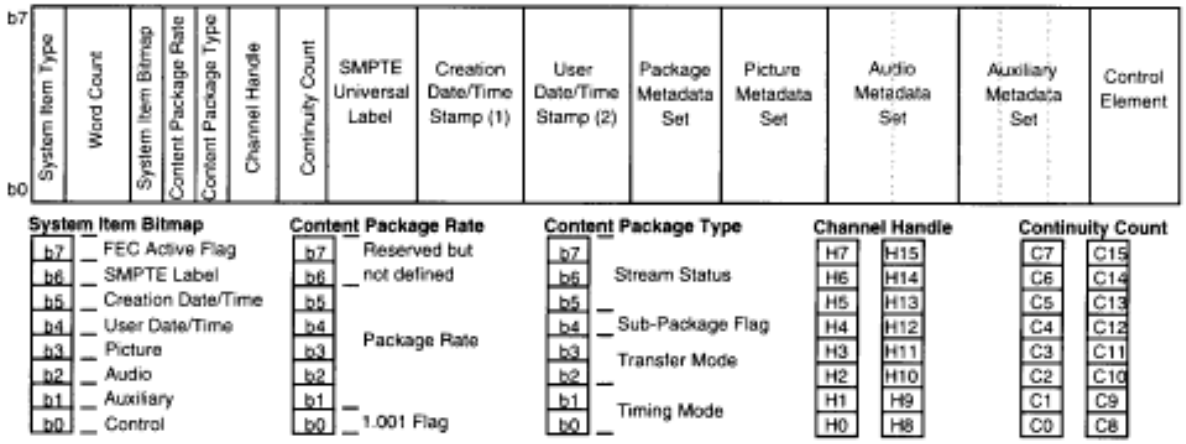

Figure 7 – System item structure

7.1 System item bitmap

The system item bitmap word is 1 byte in which bits b0-b7 have a default value of 0 indicating no data presence for the respective component. When a bit is set to 1, the corresponding part is present in the system item.

If bit b7 = 1, then forward error correction (FEC) shall be present and active;

If bit b6 = 1, then an SMPTE universal label shall be present;

If bit b5 = 1, then a creation date/time stamp shall be present;

If bit b4 = 1, then a user date/time stamp shall be present;

If bit b3 = 1, then a picture item shall be present and a picture metadata set may be present;

If bit b2 = 1, then an audio item shall be present and an audio metadata set may be present;

If bit b1 = 1, then an auxiliary item shall be present and an auxiliary metadata set may be present;

If bit b0 = 1, then a control element shall be present.

The data space associated with bits b6, b5, and b4 shall always be allocated. If bits b6, b5, or b4 are set to 0, then the value of the associated data space shall be deemed unusable.

The data space assigned to the remaining components shall be allocated if the associated bit is 1; otherwise no data space shall be allocated.

It is recommended that bit b5 be set to 1, and that a creation date/time stamp be entered correctly.

In the case of bits b3, b2, and b1, setting to 0 means that neither the associated item nor its metadata are present. When set to 1, the associated item is present and a metadata set is present. An item with no metadata will have a metadata set which indicates no metadata content.

7.2 Content package rate

The content package rate word is 1 byte which shall be used to identify the rate of content packages in baseline operation. It shall not be used to identify the frame rate of the SDTI transport structure.

Bits b7 and b6 are not defined, but are reserved for future use.

Bits b5 to b1 shall define the content package rate per second in baseline operation. These 4 bits identify 16 states defined as follows:

0 = undefined value

1 = 24	2 = 25	3 = 30
4 = 48	5 = 50	6 = 60
7 = 72	8 = 75	9 = 90
10 = 96	11 = 100	12 = 120

13-31 = reserved, but not defined.

Bit b0 identifies whether the content package rate is an exact value or offset by a factor of 1.001 and has the following values:

– 0 if the package rate is exact;

– 1 if the package rate is reduced by a factor of 1.001.

7.3 Content package type

The content package type word is 1 byte which shall be used to identify key aspects of the type and status of content package transfer.

Bits b7 to b5 shall define the position of the current content package in a stream of content packages. These 3 bits identify eight stream states as follows:

0 = the content package position in a stream is undefined;

1 = the content package is a stream head package which is any package which precedes the stream start package (e.g., preroll packages);

2 = the content package is a stream start package which is the first package of a stream;

3 = the content package is a midstream package which is any package between the stream start and stream end packages;

4 = the content package is a stream end package which is the last package of a stream;

5 = the content package is a stream tail package which is any package which follows the stream end package (e.g., postroll packages);

6 = the content package is both a stream start package and a stream end package signifying a stream of length 1;

7 = reserved but undefined.

Bit b4 shall be default 0 for all modes except low-latency transfer mode. The operation of this bit is described in 9.3.

Bits b3 and b2 identify the content package transfer mode. These 2 bits identify four states defined as follows:

0 = synchronous transfer mode: The package rate is temporally locked to the transport rate and each package is spatially locked in the defined position set by the content package timing mode;

1 = isochronous transfer mode: The package rate is not temporally locked to the transport rate, but each package is spatially locked in the defined position set by the content package timing mode;

2 = asynchronous transfer mode: The package rate is not temporally locked to the transport rate and packages are not spatially locked in defined positions;

3 = low-latency mode: Content packages are divided into subpackages and distributed throughout the transport frame period.

Bits b1 and b0 identify the content package timing mode. The timing modes are only valid for synchronous and isochronous transfer modes. These 2 bits identify four states defined as follows:

0 = normal timing mode: One content package starting on the defined first field start position;

1 = advanced timing mode: One content package starting on the defined second field start position;

2 = dual timing mode: Two content packages starting on the defined first and second field start positions;

3 = reserved but not defined.

Table 1 identifies the valid combinations of timing and transfer modes together with the clauses where they are described.

NOTE – Although all receivers shall be capable of receiving the structure of content packages in baseline operation, not all timing or transfer modes may be supported by all receivers. Even though a receiver may be capable of receiving special transfer modes, the decoder may not be able to manage correctly the decoded data to a satisfactory level as it may require special processing operations. The range of content package timing and transfer modes supported by a receiver/decoder is defined in an associated decoder template document.

Table 1 – Application of content package timing and transfer modes

	Transfer mode			
Timing mode	Synchronous	Isochronous	Asynchronous	Low-latency
Normal timing	Baseline operation	9.1.1	9.2	9.3
Advance timing	8.1	9.1.2		
Dual timing	8.2	9.1.3		
Reserved	N/A	N/A		

7.4 Channel handle number

The channel handle word consists of 2 bytes allowing a number in the range 0 to 65535 (bits H15 to H0 in figure 7). This number shall be used to distinguish different content packages having the same source and destination addresses and sent over the same SDTI link. The default value is 0000_h for single-channel transfers.

A channel handle value other than zero shall not be used until a standard or recommended practice is approved for its consistent application.

7.5 Continuity count

The continuity count word consists of 2 bytes allowing a number to be created by a modulo 65536 counter (bits C15 to C0 in figure 7). The continuity count shall increment by 1 for each newly transmitted content package with the same SDTI source and destination addresses. The continuity count may be used to detect whether the content package sequence has been broken by an operation such as a routing switch.

7.6 SMPTE universal label

The SMPTE universal label is a unique code which identifies the bitstream as a content package together with an associated decoder template.

The format comprises 16 bytes of the universal label as defined in ANSI/SMPTE 298M. If an SMPTE universal label is not present, then all 16 bytes shall be filled with 00_h and bit b6 of the system itembitmap word (7.1) shall be set to zero.

The SDTI content package has a base label value. Any associated decoder template will identify three further words of the label which shall be defined as:

– the decoder template class value;
– the decoder template type value; and
– the decoder template extension value.

The full SMPTE universal label string of 16 bytes shall be used to identify this as an SDTI content package format specified to a decoder template having the value indicated by the string values given in table 2.

Table 2 – Specification of the content package label

Byte No.	Description	Value (hex)
1	Object identifier	06_h
2	Label size	$0E_h$
3	ISO organization	$2B_h$
4	Designation: SMPTE	34_h
5	Registry: Labels	04_h
6	Labels category: Interchange	01_h
7	Labels registry	01_h
8	Registry version	01_h
9	Wrapper labels	01_h
10	Simple wrapper labels	01_h
11	Standard: CP	01_h
12	CP version	01_h
13	Template class	XX_h
14	Template type	XX_h
15	Template extension	XX_h
16	Zero fill	00_h

This label will allow a receiver/decoder to identify the content package template.

The template class value refers to an associated decoder template document and the template type value refers to the clause within an associated decoder template document. The template extension value is 00_h where the encoded content package lies within the limits defined by the template type. A template extension value of 01_h defines that the encoded content package can be received by the specified decoder, but that there are additional backwards compatible extensions. This facility is provided where decoders have capabilities beyond the defined template type, but do not meet the capabilities defined by any other template type.

7.7 Creation and user date/time stamps

The date/time metadata components are a creation date/time stamp and a user date/time stamp where each applies to the content package timing rather than the timing of individual items or elements.

Each date/time stamp metadata has a fixed 17-byte allocation which shall always be assigned. The date/time stamp metadata type is identified by the first byte and the remaining 16 bytes shall contain the date/time stamp data. The date/time stamp metadata type and data format are defined in SMPTE 331M. If bit b5 or b4 of the system item bitmap word (7.1) is set to zero, then all 17 bytes of the associated date/time stamp shall be set to 00_h. This creates a special case where a metadata type has a value of 00_h.

It is recommended that the creation date/time stamp records the date and time of content package creation. Once set, the creation date/time stamp shall not be modified in subsequent operations.

The user date/time stamp shall be used as defined by the application requirements. Space shall be allocated for the user date/time stamp although insertion of the stamp value is optional.

Note that it is possible (and likely) that some or all of the elements of the content package may have their own date/time stamp data.

7.8 Package, picture, audio and auxiliary metadata sets

The first metadata set shall be the package metadata set. The package metadata set shall contain metadata pertaining to the content package as a whole, such as, for example, a program title.

The package metadata set shall be followed by picture, audio, and auxiliary metadata sets which shall follow the order of picture, audio, and auxiliary items in the content package. Picture, audio, and auxiliary metadata sets shall be present only if the associated picture, audio, and auxiliary item is present in the content package as indicated by bits b3, b2, and b1 of the system item bitmap word. It is recommended that metadata blocks associated with any element appear in the same order as the element within the item. There may be more than one metadata type associated with each element type.

Each metadata set starts with a 1-byte metadata count which defines the number of metadata blocks in the set. A metadata count value of 00_h is a valid value and indicates that a metadata set has no metadata blocks. In this case, the metadata set is only one word long. Any metadata set shall consist of at least one word defining the metadata count value.

For coding efficiency, there is no word count value to define the length of a metadata set. Figure 8 illustrates the structure of package, picture, audio, and auxiliary metadata sets.

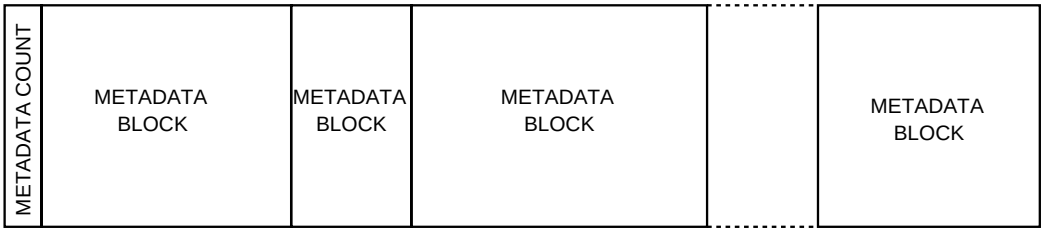

Figure 8 – Structure for metadata sets

7.8.1 Structures for package, picture, audio, and auxiliary metadata blocks

All metadata blocks shall consist of a 1-byte metadata type word, followed by a 2-byte word count word and completed by the metadata.

A metadata type value of 00_h is not permitted. Nonzero values of metadata type are defined in SMPTE 331M.

The word count value identifies the length of the metadata in bytes and shall be entered correctly.

All metadata blocks associated with any one picture, audio, and auxiliary element are grouped in sequence and each sequence is immediately preceded by a metadata link item to indicate the element to which the sequence refers. The metadata link item shall consist of a 1-byte metadata type word followed immediately by the element type and element number to which the sequence of metadata blocks refers. The element type and element number provide an unambiguous link between the metadata block sequence and the associated element.

The left side of figure 9 illustrates the structure of the metadata link item. The common structure for package, picture, audio, and auxiliary metadata blocks is illustrated on the right side.

7.9 Control element

The control element shall comprise a 1-byte element type value followed by a 4-byte word count and the control element data. The 4-byte word count value shall be entered correctly. The control element type and data format shall be as defined in SMPTE 331M.

The presence of a control element shall be as specified in an associated decoder template document.

8 Special timing modes

The modes in this clause apply only to the synchronous transfer mode.

8.1 Advanced timing mode

The advanced timing mode defines the content package timing to be advanced by 1 field. This mode can be used in certain operations to reduce system timing. Figure 10 illustrates the timing of this mode.

The content package start lines for 525/60 and 625/50 SDTI transports are defined as follows:

	525/60	625/50
Content package start:	Line 276	Line 322

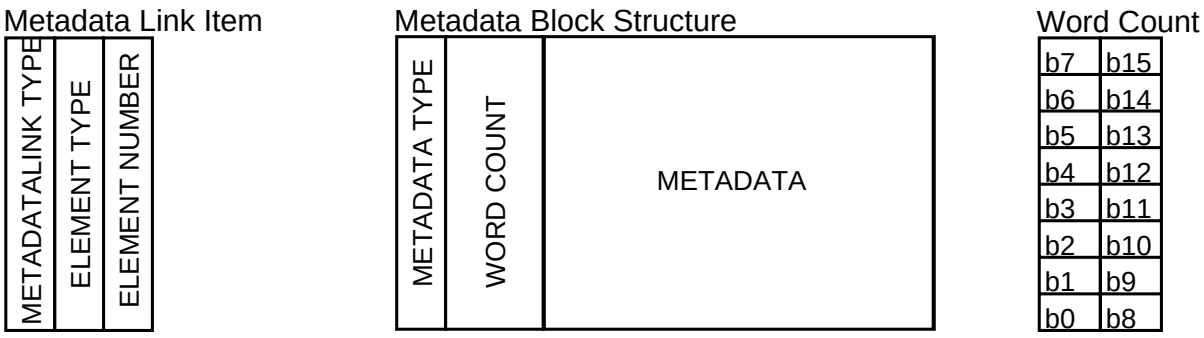

Figure 9 – Structures for the metadata link item and metadata blocks

SMPTE 326M-2000

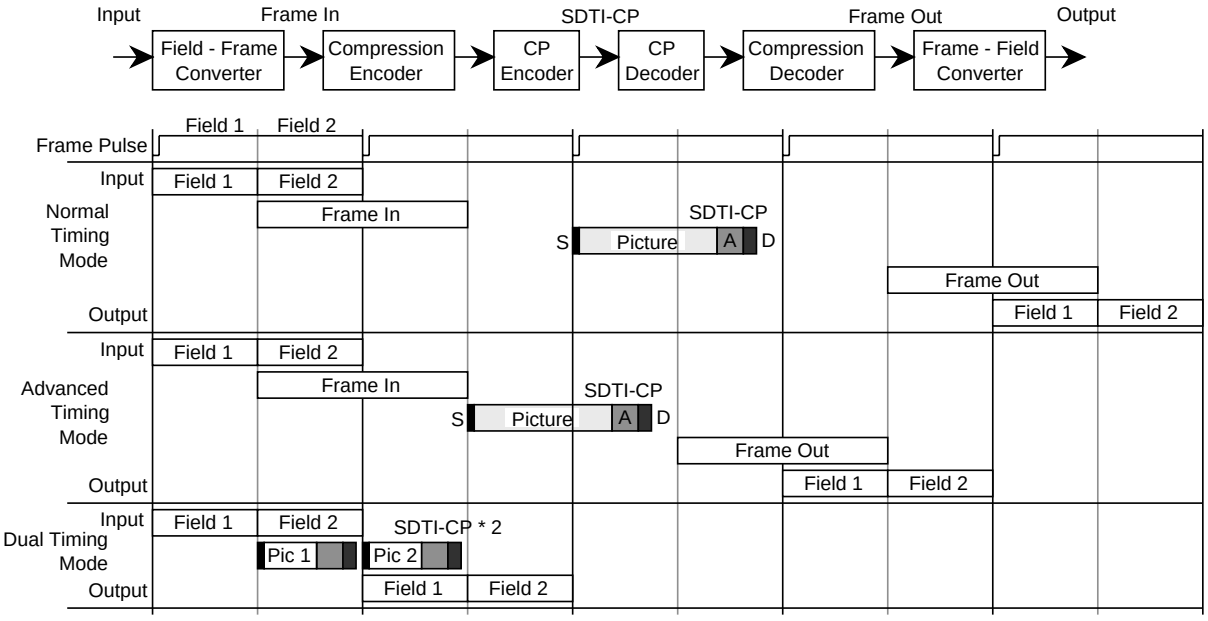

Figure 10 – Example timing diagrams for normal, advanced and dual timing modes

An associated decoder template document defines the capability to support the advanced timing mode.

8.2 Dual timing mode

The content package optionally may represent the contents of either a field of interlaced scanned picture data, or a frame of progressively scanned picture data (at 50-Hz or 60-Hz frame rate). This representation is called the dual timing mode. The general arrangement of system, picture, audio, and auxiliary items in dual timing mode is shown in figure 11.

For the dual timing mode, there are two content packages per frame with each occupying a different field and starting on the following lines:

	525/60	625/50
First field content package start:	Line 13	Line 9
Second field content package start:	Line 276	Line 322

A timing diagram of this mode is shown in figure 10.

All receivers shall be able to receive the content package on the first field of the SDTI. However, reception of the content package on the second field may not be supported by all receivers. An associated decoder template document defines the capability to fully support the dual timing mode.

8.3 Timing diagram for timing modes

The normal, advanced, and dual timing modes do not require decoder buffer delay metadata as they are defined by the synchronous nature of the SDTI transport. The timing diagram of figure 10 gives examples of the timing of the uncompressed source, content package bitstream, and decoded uncompressed outputs for all three modes, thus providing designers with a means of defining the required decoder buffer size.

The timing alignments show the minimum possible delays in each timing mode. Larger delays will be typically encountered where the input may be an integer number of frames earlier and the output an integer number of frames later than shown in figure 10.

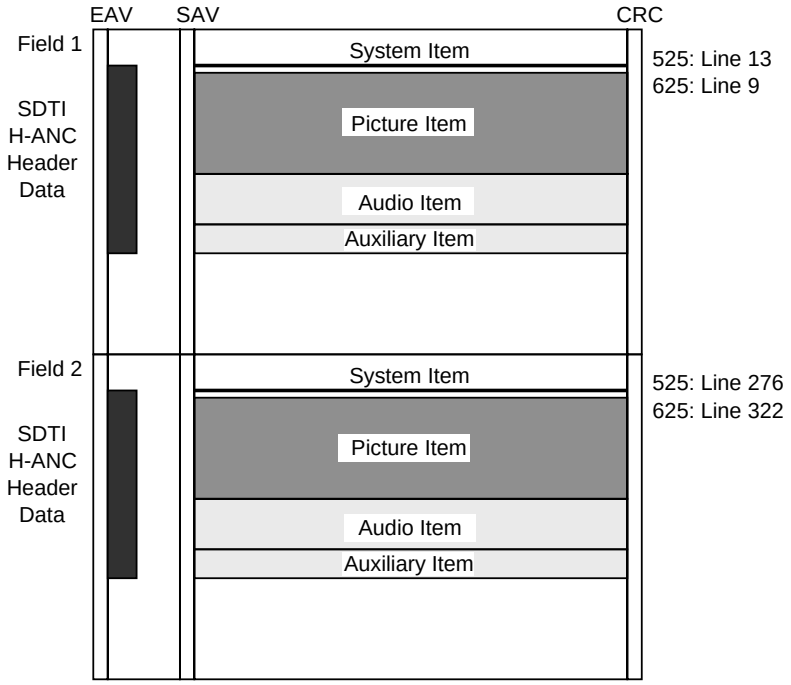

Figure 11 – Arrangement of items for dual timing mode content packages

9 Special transfer modes

When content packages are transmitted synchronous to the SDTI frame rate, the start line is defined by the timing mode. This standard allows content packages to be transmitted slower and faster than the SDTI frame rate. Furthermore, there is the capability for content packages containing different program material from the same or different source equipment to be multiplexed onto the same SDTI link. This standard provides the following points to aid correct operation of such modes:

– With the exception of the low-latency mode, each content package shall be transmitted as a continuous sequence of SDTI lines. No content packages shall suffer breakup as a result of multiplexing or any other operation except in the case of the low-latency mode.

– Demultiplexing of multichannel sources can be achieved by using the SDTI source and destination addresses. Different content package sequences multiplexed onto the same SDTI link and having identical source and destination addresses shall have different channel handle numbers.

In these special transfer modes, decoders must provide sufficient buffering for the intended application area and this buffering will typically be greater than that required for synchronous transfers. It should also be noted that these special transfer modes may require special decoding processes to ensure adequate signal recovery to be able to present viewable pictures and acceptable sound quality.

9.1 Isochronous transfer mode

Isochronous transfers are defined as the transfer of content packages at nonnormal transfer rates, such as slower or faster than real time, while retaining their lock to the spatial alignment position; i.e., the system item always appears on the defined start line. It should be noted that in the case of transfer rates which are changing (e.g., during a tape speed transition), transient conditions may occur which do not meet the specifications set in the remainder of this clause. Decoders preferably shall have sufficient buffering to be able to manage transient states without undue perturbations in the buffer output. In the descriptions following, the number N represents the ratio of the transfer speed to real time transfer in baseline operation.

Thus, a value of N=2 means that the transfer speed is twice the real time transfer speed.

9.1.1 Normal timing mode

In slower-than-real-time transfer, the start of each content package shall be at every 1/Nth frame on the defined line for normal timing mode. Where N is not an integer value, it is preferred that the distribution of content packages over time be as even as possible with the number of frames between content packages differing by less than one for any given transfer speed up to unity.

In faster-than-real-time transfer, the start of every Nth content package shall be on the defined line for normal timing mode followed immediately by the next N-1 content package. Where N is not an integer value, it is preferred that the number of N content packages in a frame period be as evenly distributed as possible with the number differing by less than one for any given transfer speed above unity.

9.1.2 Advanced timing mode

In slower-than-real-time transfer, the start of each content package shall be at every 1/Nth frame on the defined line for advanced timing mode. Where N is not an integer value, it is preferred that the distribution of content packages over time be as even as possible with the number of frames between content packages differing by less than one for any given transfer speed up to unity.

In faster-than-real-time transfer, the start of every Nth content package shall be on the defined line for advanced timing mode followed immediately by the next N-1 content package. Where N is not an integer value, it is preferred that the number of N content packages in a frame period be as evenly distributed as possible with the number differing by less than one for any given transfer speed above unity.

The timing modes described in this clause are illustrated in outline form in figure 12.

9.1.3 Dual timing mode

In slower-than-real-time transfer, the start of each content package shall be at every 1/Nth field on the defined line for dual timing mode. Where N is not an integer value, it is preferred that the distribution of content packages over time be as even as possible with the number of fields between content packages differing by less than one for any given transfer speed up to unity.

In faster-than-real-time transfer, the start of every Nth content package shall be on the defined line for dual timing mode followed immediately by the next N-1 content package. Where N is not an integer value, it is preferred that the number of N content packages in a field period be as evenly distributed as possible with the number differing by less than one for any given transfer speed above unity.

The timing modes described in this clause are illustrated in outline form in figure 12.

9.2 Asynchronous transfer mode

Asynchronous transfers are used to transfer content packages in the correct sequence, but with an arbitrary timing not locked to frame or field boundaries. Such transfers require a specialist buffer or a storage medium capable of receiving packages with an undefined separating space between packages. By the nature of its definition, an asychronous transfer overrides the relevance of the timing modes. Figure 12 illustrates the use of the asychronous transfer mode.

The ability of a receiver to decode the structure of content packages in the asychronous transfer mode is defined in an associated decoder template document.

Asynchronous transfer mode also permits the application of nonreal time transfers of content packages whose size exceeds an SDTI frame length. Any application requiring the transfer of content packages whose size exceeds the length of an SDTI frame shall be defined by a specific decoder template recommended practice for that application.

9.3 Low-latency transfer mode

To reduce system delay, content packages may be divided into a sequence of subpackages. Each subpackage shall contain a system item together with one or more subitems. Subitems are defined as per an item, but have a fraction of the frame based data content associated with baseline operation.

The system item of the first subpackage shall be as defined for baseline operation and shall contain any metadata as appropriate for the content package as a whole.

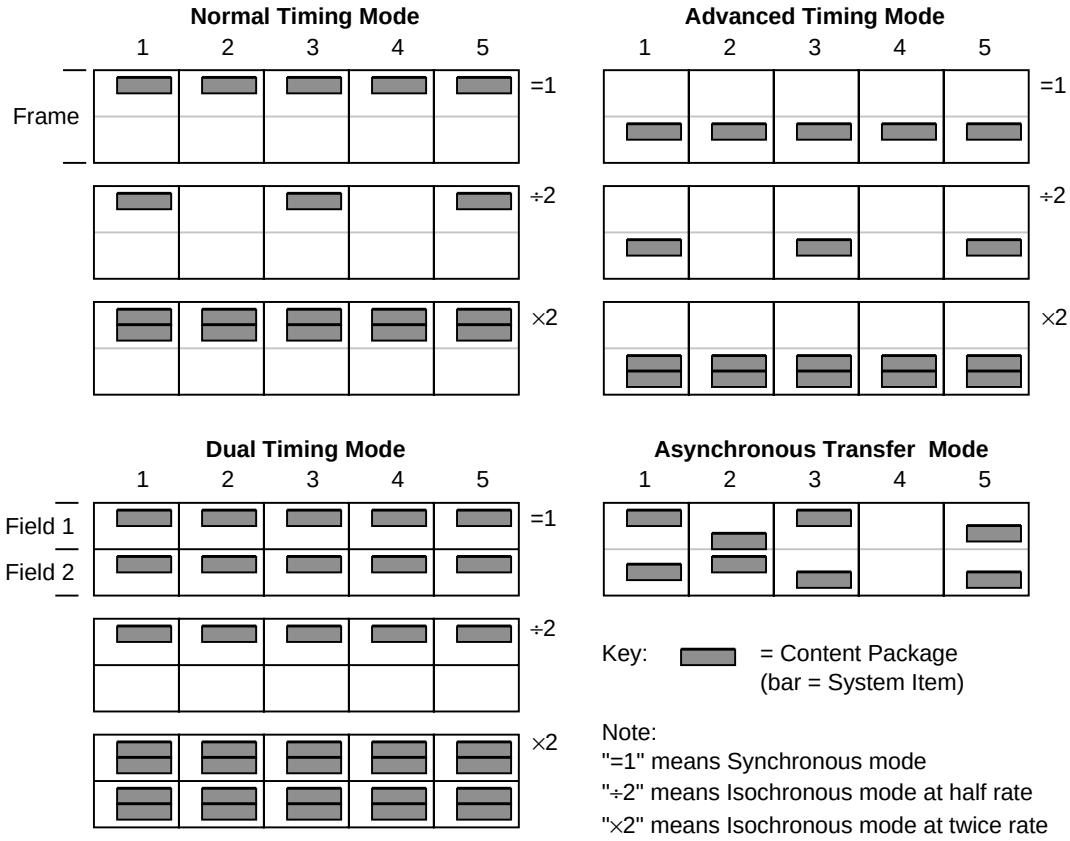

Figure 12 – Illustration of timing and transfer modes

Bit b4 of the content package type word in the system item shall be set to 0 of the first subpackage.

The system item of subsequent subpackages shall contain at least the first 7 header bytes from the system item bitmap to the continuity count inclusive. The remaining parts of the system item are optional, but shall be ignored by a decoder if present.

Bit b4 of the content package type word shall be set to 1 for the system item of these subsequent subpackages. All other data in the 6 header bytes following the system item bitmap word shall be identical to the corresponding words in the system item of the first subpackage.

9.3.1 Buffer delay

To support low-latency transfer mode, a decoder delay metadata item may explicitly define the buffer delay of the receiver. This metadata item is specified in SMPTE 331M.

The buffer delay operation is initiated by the reception of the first subitem sample written into the item buffer. The buffer read operation is set to start reading the buffer data when the decoder delay time has been equaled or exceeded. The decoder delay time is specified in units of a 90-kHz clock.

An example of low-latency transfer mode subpackages is illustrated in figure 13.

The ability of a receiver to receive and parse content packages in the low-latency transfer mode is defined in an associated decoder template document. If an associated decoder template document allows low-latency transfer mode, it may also restrict the number and size of subitems and the order of subitems in a subpackage.

10 Error correction

Forward error correction (FEC) may optionally be added to the payload area where extra security is required.

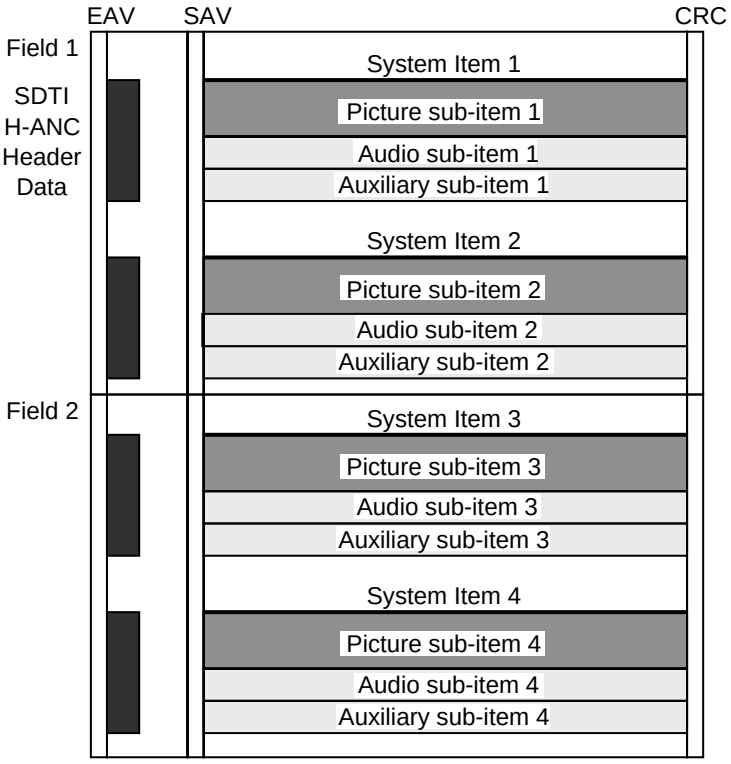

Figure 13 – Illustration of the low-latency transfer mode

A case where FEC might be added is where the payload data have no in-built redundancy; e.g., long GOP MPEG-2 coded pictures and vital IT data files.

A case where FEC need not be added is where the payload data have some residual redundancy; e.g., uncompressed or lightly compressed picture or audio signals.

10.1 Error correction format

FEC shall only be applied to the least significant 8 bits of the 10-bit interface which have previously been identified as bits b0 to b7. The payload data area shall be grouped into 240 word FEC blocks as illustrated in figure 14. The first 234 words of each FEC block shall be used for content package data and the last 6 words are used for the Reed-Solomon FEC. For all 240 words of the FEC block, bits b8 and b9 shall be as defined by SMPTE 305M.

FEC blocks will apply across the whole of any line associated with the content package data. Thus, for an active line length of 1440 words, the available data space for content package data would be reduced to 1404 words. Any line associated with the content package will have FEC blocks for the whole line length regardless of whether the content package data occupies all the available data space.

The error correction is defined as a Reed-Solomon R-S (240, 234, T=3) shortened code from the original R-S (255, 249, T=3) code.

The R-S code generator polynomial shall be:

$$R\text{-}S(x) = (x \oplus a^0).(x \oplus a^1).(x \oplus a^2).(x \oplus a^3).(x \oplus a^4).(x \oplus a^5)$$

where a is defined by the Galois field GF(256) generator polynomial:

$$GF(x) = x^8 \oplus x^4 \oplus x^3 \oplus x^2 \oplus 1.$$

Each word of a 240-word FEC block shall have bits b8 and b9 set according to SMPTE 305M.

Figure 14 illustrates the application of FEC to the payload area.

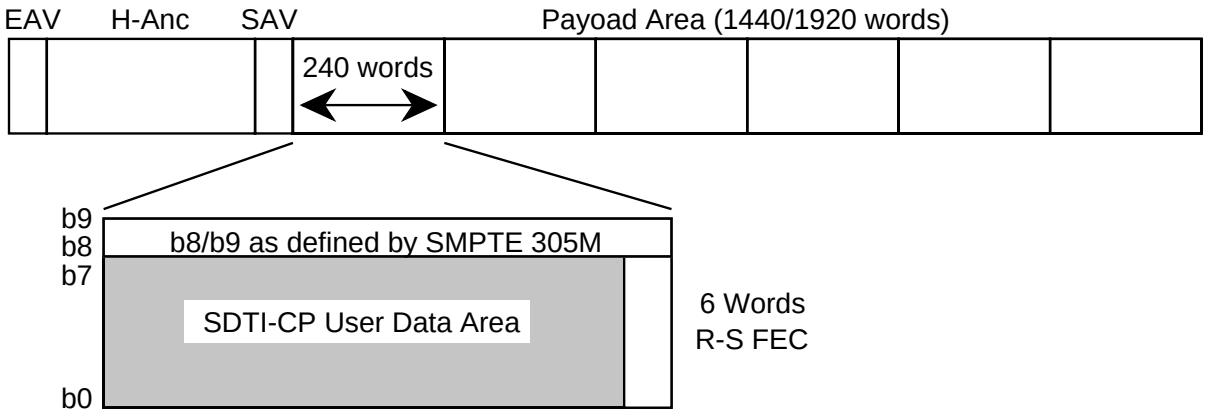

Figure 14 – Specification of the optional SDTI-CP FEC

10.2 FEC Identification

Bit b7 of the system item bitmap word in the system item header is used to flag the application of FEC in the payload area of all lines used in the associated content package. In the case of low-latency mode, this also applies to all associated subpackages. Where a line has FEC applied, the FEC shall apply to all the FEC blocks in that line.

If the SDTI CRC and the SDTI-CP FEC are both active, the last two words of the last FEC block in any line may interfere with the CRC words. To pevent this condition, the following rules shall be applied to any line carrying content package data:

– If the FEC is active, then the SDTI CRC shall not be used and the last 2 bytes of the last FEC block on the line will overwrite the CRC location. The payload CRC flag of SMPTE 305M shall be set to 00_h.

– If the FEC is not active, then the line may apply the SMPTE 305M CRC and set the payload CRC flag accordingly.

10.3 Word count values

The word count value of the system, picture, audio, and auxiliary items are defined by the variable block structure of SMPTE 305M. Where FEC is applied, the word count values of items shall include all FEC words which are contained by a variable block in order to comply with SMPTE 305M.

In the case of word counts defined solely by this standard, the values shall represent the block sizes without including any FEC words which may be present.

Annex A (normative)
Syntax of the content package structure as pseudo-code

The following pseudo-code illustrates the structure of the SDTI-CP fomat in baseline operation only. The values in square brackets indicate the length of the variable in bytes; [var] indicates a variable length packet.

```
For (content_package=start; content_package<=end;
  content-package++)
{
  System item()
  {
    Separator [1]
    System item type [1]
    Word count [4]

    System item bitmap [1]
      FEC_flag = 80h and system item bitmap
      SMPTE_flag = 40h and system item bitmap
      Creation_TC_flag = 20h and system item bitmap
      User_TC_flag = 10h and system item bitmap
      Picture_flag = 08h and system item bitmap
      Audio_flag = 04h and system item bitmap
      Aux_flag = 02h and system item bitmap
      Control_flag = 01h and system item bitmap

    Content package rate [1]
    Content package type [1]
    Channel handle [2]
    Continuity count [2]

    If (SMPTE_flag)
      SMPTE universal label [16]
    Else
      Null data [16]

    If (Creation_TC_flag)
      Type [1]
      Creation date/time code [16]
    Else
      Null data [17]

    If (user_TC_flag)
      Type [1]
      User date/time code [16]
    Else
      Null data [17]

    Package metadata ()
    {
      Metadata_count [1]

      for (metadata=0; metadata<metadata_count; metadata++)
      {
        Metadata type [1]
        Metadata word count [2]

        Package metadata [var]
      }
    }

    If (picture_flag)
    {
      Picture metadata ()
      {
        Metadata_count [1]

      for (metadata=0; metadata<metadata_count; metadata++)
      {

        If (metadata_sequence_start)
        {
          Metadata link type [1]
          Element type [1]
          Element number [1]
        }

        Metadata type [1]
        Metadata word count [2]

        Picture metadata [var]
      }
    }
  }

  If (audio_flag)
  {
    Audio metadata ()
    {
      Metadata_count [1]

      for (metadata=0; metadata<metadata_count; metadata++)
      {
        If (metadata_sequence_start)
        {
          Metadata link type [1]
          Element type [1]
          Element number [1]
        }

        Metadata type [1]
        Metadata word count [2]

        Audio metadata [var]
      }
    }
  }

  If (aux_flag)
  {
    Aux metadata ()
    {
      Metadata_count [1]

      for (metadata=0; metadata<metadata_count; metadata++)
      {
        If (metadata_sequence_start)
        {
          Metadata link type [1]
          Element type [1]
          Element number [1]
        }

        Metadata type [1]
        Metadata word count [2]

        Aux metadata [var]
      }
```

```
    }
  }

  If   (control_flag)
  {
    Control  element  ()
    {
      Element  type  [1]
      Element  word  count  [4]
      Element  packet  [var]
      }
    }

    End  code  [1]
  }

  If   (picture_flag)
  {
    Picture   item
    {
      Separator   [1]
      Picture  item  type  [1]
      Item  word  count  [4]
      element_count   [1]

      for  (element=0;  element<element_count; element++)
      {
        Element type [1]
        Element word count [4]
        Element number [1]

        Picture element data [var]
      }

      End code [1]
    }
  }

  If (audio_flag)
  {
    Audio item
    {
      Separator [1]
      Audio item type [1]
      Item word count [4]
      element_count [1]

      for (element=0; element<element_count; element++)
      {
        Element type [1]
        Element word count [4]
        Element number [1]

        Audio element data [var]
      }

      End code [1]
    }
  }

  If (aux_flag)
  {
    Aux item
    {
      Separator [1]
      Aux item type [1]
      Item word count [4]
      Element_count [1]

      for (element=0; element<element_count; element++)
      {
        Element type [1]
        Element word count [4]
        Element number [1]

        Aux element data [var]
      }

      End code [1]
    }
  }
}
```

Annex B (informative)
Bibliography

ANSI/SMPTE 259M-1997, Television — 10-Bit 4:2:2 Component and $4f_{sc}$ Composite Digital Signals — Serial Digital Interface

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

ISO/IEC 13818-2:1996, Information Technology — Generic Coding of Moving Pictures and Associated Audio Information: Video

ISO/IEC 13818-2:1996, Amendment 2:1997 (MPEG-2, 4:2:2P@ML)

EBU/SMPTE Task Force for Harmonized Standards for the Exchange of Program Material as Bitstreams, Final Report: Analyses and Results, July 1998, SMPTE J. 107(9):603-815; 1998 September

SMPTE 331M-2000

SMPTE STANDARD

for Television — Element and Metadata Definitions for the SDTI-CP

1 Scope

This standard specifies the formats of the elements and metadata used by the SDTI content package format standard (SDTI-CP), SMPTE 326M.

This standard defines element and metadata formats where they are simply specified or where a publicly available reference is accessible. It is not intended that this standard provide detailed specifications for complex formats which may have broader application.

2 Normative references

The following standards contain provisions which, through reference in this text, constitute provisions of this standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this standard are encouraged to investigate the possibility of applying the most recent edition of the standards indicated below.

AES3-1992, Digital Audio Engineering — Serial Transmission Format for Two-Channel Linearly Represented Digital Audio Data

SMPTE 12M-1999, Television, Audio and Film — Time and Control Code

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

SMPTE 309M-1999, Television — Transmission of Date and Time-Zone Information in Binary Groups of Time and Control Code

SMPTE 312M-1999, Television — Splice Points for MPEG-2 Transport Streams

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

SMPTE 328M-2000, Television — MPEG-2 Video Elementary Stream Editing Information

SMPTE RP 186-1995, Video Index Information Coding for 525- and 625-Line Television Systems

ISO/IEC 10918-1:1994, Information Technology — Digital Compression and Coding of Continuous-Tone Still Images: Requirements and Guidelines

ISO/IEC 10918-3:1997, Information Technology — Digital Compression and Coding of Continuous-Tone Still Images: Extensions

ISO/IEC 13818-2:1996, Information Technology — Generic Coding of Moving Pictures and Associated Audio Information: Video

Amendment 2:1997 to ISO/IEC 13818-2:1996, (MPEG-2, 4:2:2P@ML)

3 Introduction

Each type of element and metadata listed below includes an element or metadata type value. This type value is not related to the data type values specified in SMPTE 305M. The SDTI data type defines an item type, whereas this standard specifics element and metadata types.

The ranges of element and metadata type values for each item shall be as follows:

– Picture item: Element type range = 01_h to $0F_h$ inclusive;

Approved
January 31, 2000

– Audio item: Element type range = 10_h to $1F_h$ inclusive;

– Auxiliary item: Element type range = 20_h to 77_h inclusive;

– System item: Element type range = 78_h to $7F_h$ inclusive;

– System item: Metadata type range = 80_h to FF_h inclusive.

A type value of 00_h is not a valid value for either element or metadata.

4 System elements

4.1 Control code

Type value = 78_h

Reserved but not yet defined.

5 Picture elements

5.1 MPEG-2 picture element

Type value = 01_h

The MPEG-2 picture element is the MPEG-2 video elementary stream (V-ES) of any profile or level.

The key start codes for MPEG-2 video elementary streams are:

– Sequence header (S): 00, 00, 01, B3;
– GOP header (G): 00, 00, 01, B8;
– Picture header (F): 00, 00, 01, 00;
– Sequence end code (E): 00, 00, 01, B7.

An example V-ES bit stream (according to SMPTE RP 204) is shown in figure 1. The MPEG-2 V-ES bit stream is simply formatted into a data block as indicated in the figure. No other data are required.

It is recommended that the V-ES be compliant with SMPTE 328M. The following list of points summarizes the recommendations of that standard for the repetition of MPEG-2 GOP and sequence header information:

– If the picture to be formatted is not an I-picture, then the data from the picture header code, up to but not including either the next GOP or picture header, are formatted into a block;

– If the picture to be formatted is an I-picture, then the data from the sequence, GOP, and picture headers, up to but not including either the next GOP or picture header, are formatted into a block;

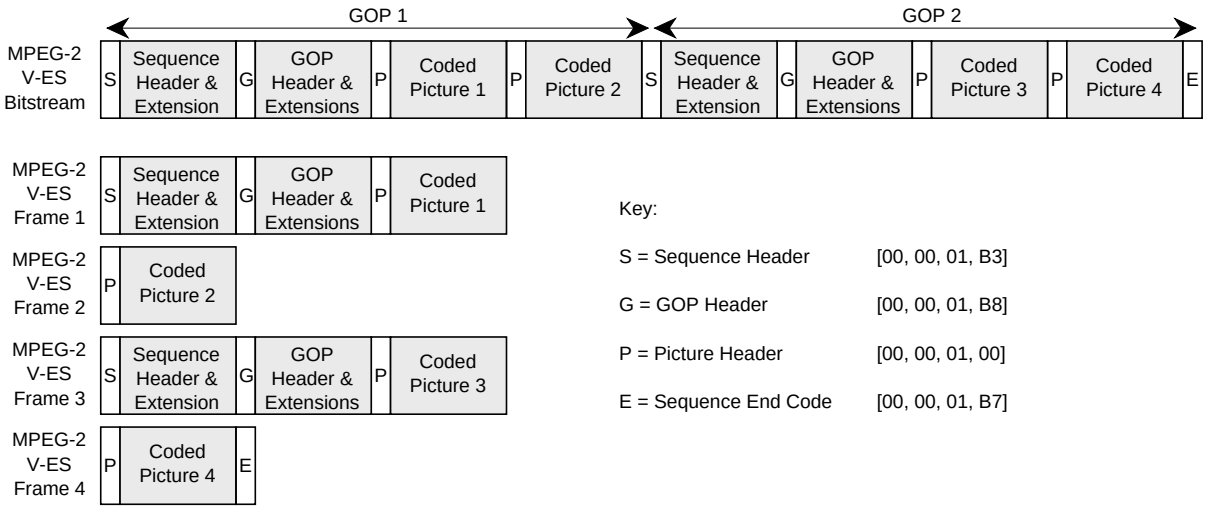

Figure 1 – Example formatting of a V-ES into SDTI-CP element frames

– It is recommended that the sequence header information be repeated at each I-picture with the information placed immediately prior to the GOP header information. Thus, information about the sequence is readily available following any editing process. If sequence header information were not repeated so frequently, then edit processes may easily remove this information making downstream processing more difficult or even impossible;

– A sequence end code shall be retained with the end of the last picture in the sequence. After editing, a new sequence end code shall be added to the end of a sequence if it does not already exist.

The sequence of pictures is as per the MPEG-2 picture bit stream including any discontinuity which may result from the use of B-pictures. Any picture stream timing metadata in the system item shall reflect the decoded picture display sequence and thus the timing may appear discontinuous in the event of B-frames.

The byte alignment of both MPEG-2 and SDTI-CP is identical and this alignment is maintained in this standard. However, it should be noted that the bit stream orders of MPEG-2 and SDI differ in that the MPEG-2 bit stream is MSB first, whereas the SDI bit stream is LSB first.

6 Audio elements

6.1 8-channel AES3 element

Type value = 10_h

Uncompressed audio formats are the preferred choice for television production use. However, the format allows for the safe carriage of all data when coded as an AES3 signal.

The data format of each channel of the 8-channel AES3 audio element is defined by the AES3 interface specification. Although the AES3 specification is limited to 2 channels, the 8-channel element is able to carry up to eight individual channels of AES3 data transparently. The I/O to the element will typically use AES3 twin-channel interfaces. Each AES3 channel may contain either linear PCM audio or data according to the AES3 specification.

The data format for an 8-channel AES3 audio element is shown in figure 2. The element data area shall contain audio samples for the period of the picture frame as close as possible.

Up to 8 channels of AES3 audio data shall be multiplexed on a word-by-word basis; i.e., the first word (W) of each channel (Ch) is multiplexed into the sequence:

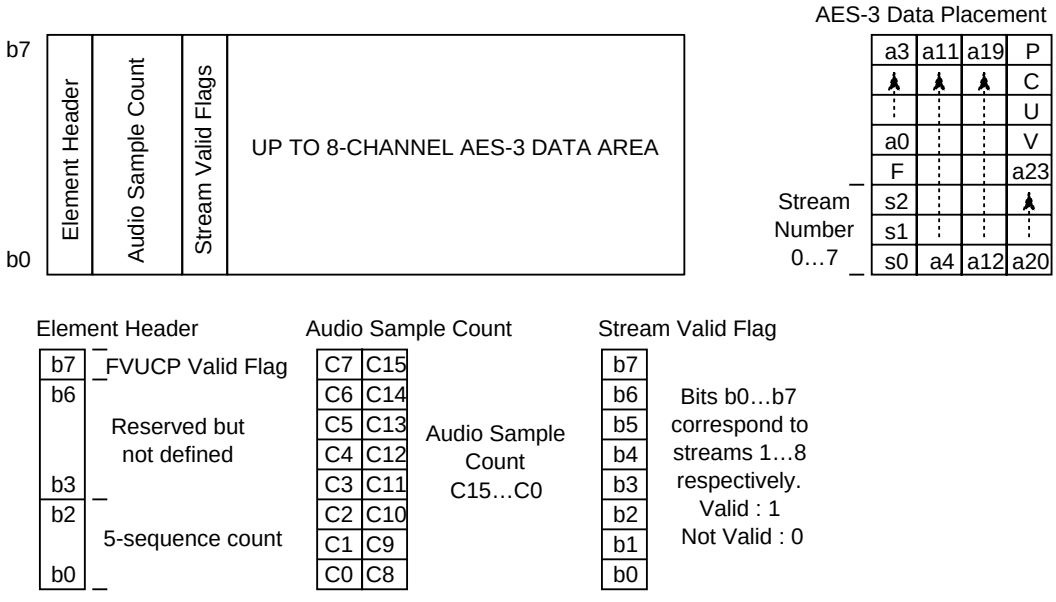

Figure 2 – Audio element format

W1 Ch1, W1 Ch2, W1 Ch3, W1 Ch4, W1 Ch5, W1 Ch6, W1 Ch7, W1 Ch8

W2 Ch1, W2 Ch2, W2 Ch3, W2 Ch4, W2 Ch5, W2 Ch6, W2 Ch7, W2 Ch8, etc.

The format of the bits in each word is defined in figure 2.

The stream number is defined by bits s2 to s0. These bits define 8 states where 0 represents stream 1 and 7 represents stream 8.

The F bit indicates the start of a new AES3 subframe. This bit is a 1 if the word is at the start of the AES3 subframe, else it is 0.

The 24 bits of the AES3 specification are directly mapped into bits a0 to a23. The V, U, C, and P bits are also directly mapped as shown in figure 2.

For the audio stream element header:

– Bit b7 indicates if the FVUCP bits are active. A value of 0 indicates that the FVUCP bits are not used. A value of 1 indicates that the FVUCP bits are valid and usable.

– Bits b6 to b3 are not defined, but reserved for future use.

– Bits b2 to b0 define a 5-sequence count. In a content package based on the 525/59.94 system, the count shall be a (modulo 5 + 1) count over the range 1 to 5. In a content package based on the 625/50 system, or any other system where the audio sample count is a consistent integer value over the content package period, the count shall be set to 0. All AES3 data channels within the same element shall have the same 5-sequence count number.

In the particular case of content packages based on 525/59.94 systems, the 5-sequence count defines one of the following sets of sample numbers per content package depending on whether it is frame or field based:

Sequence No.	30/1.001	60/1.001
1	1602	801
2	1601	801
3	1602	800
4	1601	801
5	1602	801

The audio sample count is a 16-bit count in the range 0 to 65535 and represents the number of samples in each channel. All channels within the element shall have the same audio sample count value.

The data valid flag word has 8 bits, b0 to b7, which reflect the validity status of the corresponding channels 1 to 8. A valid data flag bit is set to 1 if the channel contains a meaningful audio data, else it is set to 0. The AES3 data area shall carry data only for channels which have the channel valid bit set to 1.

7 Auxiliary elements

7.1 VBI line format

Type value = 20_h

The VBI line element carries one or more lines from the vertical blanking interval. The VBI line element has a header which identities whether the source is interlaced or progressive and a number to identify the number of VBI lines carried.

Each VBI line is created from one line of the vertical blanking interval. Each line starts with a VBI information word followed by the 8-bit words from the whole of the VBI line.

Details of the VBI line block structure are shown in figure 3.

The order of the VBI lines shall be as they are displayed on a viewing device.

For an interlaced scanned system, this shall be in the following order:

– [VBI 1, 1st field], [VBI 2, 2nd field],
– [VBI 3, 1st field], [VBI 4, 2nd field],
– [VBI 5, 1st field], [VBI 6, 2nd field].

For a progressive scanned system, this shall be in the following order:

VBI 1, VBI 2, VBI 3, VBI 4, VBI 5, VBI 6.

In the VBI header words:

– Bits N3 to N0 of the first word shall define the number of VBI lines. The allowable values shall be in the range 0 to 6;

– Bits L3 to L0 of the first word together with bits L8 to L11 of the second word form a 12-bit count value which identifies the length of the VBI lines. All VBI lines in one element shall have the same length.

In the VBI position word:

– Bits b14 to b8 of the second word and bits b7 to b0 of the first word form a line number range of 0 to 32767. The line address number shall represent an absolute line number for both interlaced and progressive line-numbering systems.

– Bit b7 of the second word (P) is set to 0 for interlaced scan and 1 for progressive scan.

A line address value of 0 means that no line number has been defined. Any line address number outside the vertical interval period for the picture scanning system is invalid and may cause unspecified effects in receiving equipment.

7.2 Ancillary data packet format

Type value = 21_h

Ancillary data packets are defined by SMPTE 291M. This format carries only the 8 LSBs of each word of the ANC data packets and removes the ADF word sequence. Where there is more than one ANC packet, they shall be packed in sequence with no padding words or gaps between the packets. Reformatting to the full 10-bit word resolution together with the addition of the ADF sequence to the head of each ANC packet is the responsibility of the output reformatting device.

Ancillary data packets carry data identification codes (DID and SDID) which identify the type of payload. The values for these codes are defined in the appropriate SMPTE standards and recommended practices.

The format of a type 2 ancillary data packet is shown in figure 4. The format may carry type 1 ancillary data by replacing the secondary DID word with the data block number.

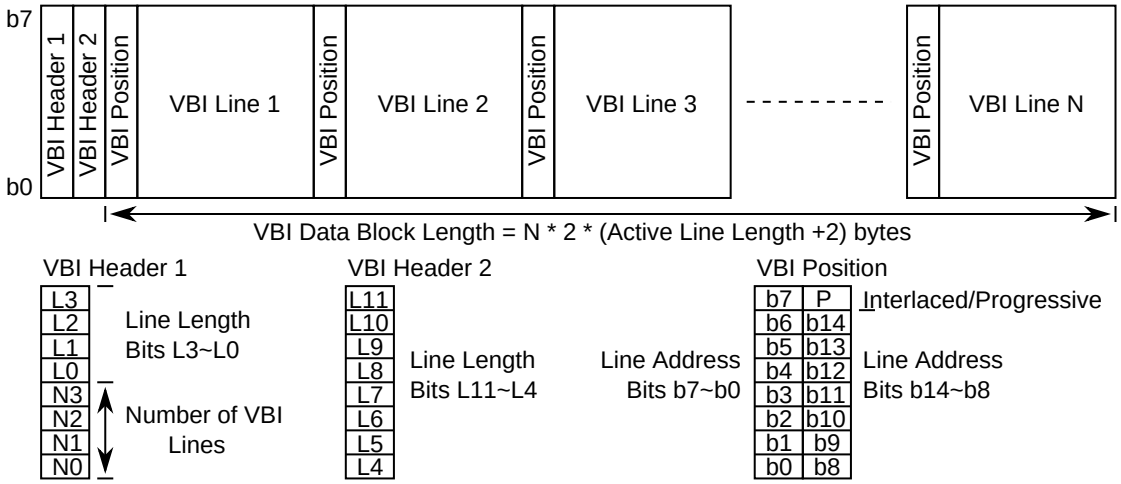

Figure 3 – VBI line format

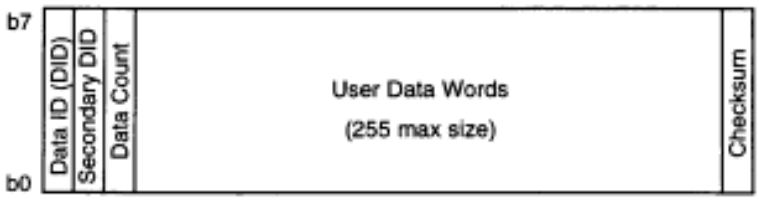

Figure 4 – Ancillary data packet format (Type 2 shown)

7.3 General data format

Type value = 22_h

The general format shown in figure 5 is used to carry all free-form data types which do not have a separate auxiliary item element type value. These data types, including those of an IT nature (Word processing files, Hypertext, etc.), may require format identification through the associated metadata in the system item.

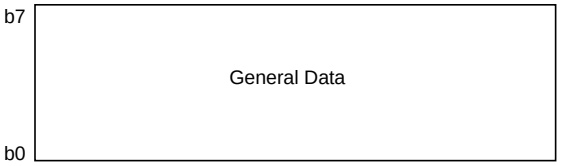

Figure 5 – General data format

The auxiliary item may contain several general data elements with the same type identifier. These are distinguished with the element number which shall match the element type and number used in any associated metadata packet.

7.4 EBU audio WAVE file format

Type value = 40_h

Reserved, but not defined.

This element type may also be placed in the audio item if it is part of a stream of files.

7.5 SPIFF (or JFIF) file format

Type value = 41_h

The JPEG compression format is as specified by ISO/IEC 10918-1.

NOTE – Only one of the following file formats should be used at the time of publication:

– The file interchange format for JPEG compression is SPIFF as specified by ISO/IEC 10918-3.

– The file interchange format for JPEG compression is the JFIF format.

At the time of writing, the open specification of the JFIF file format can be downloaded using ftp://ftp.uu.net/graphics/jpeg/jfif.ps.gz.

This element type may also be placed in the video item if it is part of a stream of files.

7.6 Tagged image file format (TIFF)

Type value = 42_h

Revision 6 as specified by Adobe Systems Incorporated, 3rd June 1992.

This element type may also be placed in the video item.

NOTE – At the time of writing, the open specification of this file format can be accessed by a web browser with the URL: http://partners.adobe.com/supportservice/devrelations/technotes.html.

8 System metadata definitions

The following metadata definitions may be used in reference to any element in the picture, audio, or auxiliary items.

NOTE – Each metadata definition has both a local identifier defined by the type value and a global identifier which defines the place of the metadata item in the SMPTE metadata dictionary structure. Both identifiers are referencing the same metadata specification. The reason for the shortened type value used in the content package is for ease of parsing the data at the high speeds used by the SDTI transport. There is also a gain in packing density and hence simplified storage requirements on high-speed silicon. But it should be noted that any metadata item specified in this standard may be expanded to define the full K-L-V construct on which the metadata dictionary structure is based. This fully expanded K-L-V construct may then be used as a basis for the common interchange of metadata items between different applications.

The SMPTE metadata dictionary is currently under construction and the values given are tentative and may be subject to revision. Readers of this standard are encouraged to check with the published version of the dictionary to confirm the accuracy of the global key values. For this reason, the global key values are shown for information only and shall not be regarded as definitive until this standard has been updated.

Note that the first 7 words of each key value are the same and set at the value:

06_h, $0E_h$, $2B_h$, 34_h, 01_h, 01_h, 01_h.

To simplify the global key values, only the last 9 words are indicated in this standard. Thus, the first word of each global key value identifies the metadata dictionary class type.

8.1 Metadata link item definition

Local type value = 80_h

Global key value = 01_h, 02_h, 02_h, 01_h, 00_h, 00_h, 00_h, 00_h, 00_h

(Essential metadata, locators, local locators, CP locator)

This special metadata item shall be used to link subsequent metadata items to their respective elements. The format differs from the general metadata construct and consists simply of 3 bytes which shall be in the sequence:

Metadata type, element type, element number.

In any picture, audio, or auxiliary metadata set, a metadata link item shall be present immediately following any nonzero metadata count value. All metadata items following a metadata link item shall refer to the defined element until the occurrence of the next metadata link item.

8.2 SMPTE 12M time code definition

Type value = 81_h

Global key value = 05_h, 03_h, 01_h, 00_h, 00_h, 00_h, 00_h, 00_h, 00_h

(Relational, video time stamps, 12M [LSB + packed])

This metadata consists of a 16-byte field with the first 8 bytes coded with the data from the SMPTE 12M time code specification and is shown in figure 6. The last 8 bytes of the metadata format shall be null filled.

Note that the metadata is organized as LSB first to comply with SDI convention.

The order of transmission is bit b0 of the leftmost word first finishing with bit b7 of the rightmost word.

A list of abbreviated terms in figure 6 and their full names follows:

- CF flag: color frame flag;
- DF flag: drop frame flag;
- FP: Field phase (NTSC), Binary group flag 0 (PAL);
- B0: Binary group 0 (NTSC), Binary group 2 (PAL);
- B1: Binary group 2 (NTSC), Field phase (PAL);
- B2: Binary group 1.

8.3 SMPTE 309M date-time code definition

Type value = 82_h

Global key value = 05_h, 03_h, 05_h, 00_h, 00_h, 00_h, 00_h, 00_h, 00_h

(Relational, video time stamps, 309M [LSB + packed])

This metadata consists of a 16-byte field with the first 8 bytes coded according to the SMPTE 309M specification with a time, time zone, and date fields. The last 8 bytes of the metadata format shall be null filled.

The mapping of data from SMPTE 309M to the SDTI-CP metadata format is identical to that described in 8.2 (SMPTE 12M time code).

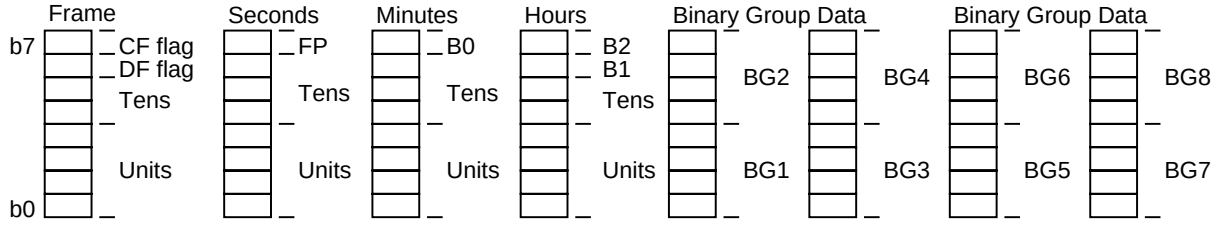

Figure 6 – Definition of the time stamp data placement

8.4 SMPTE UMID definition

Type value = 83_h

Global key value = 01_h, 01_h, $01/02/03/04_h$, 00_h, 00_h, 00_h, 00_h, 00_h , 00_h

(Essential, unique identifiers, UMID_picture/audio/data/other)

Reserved, but not yet defined.

NOTE – A UMID is an identifier for four basic essence types: picture, audio, data, and other. Thus, the UMID global key can be identified by using the essence type value from the range defined in the introduction of this standard.

8.5 MPEG-2 picture editing metadata

Type value = 84_h

Global key value = $0B_h$, 01_h, 01_h, 01_h, 00_h, 00_h, 00_h, 00_h, 00_h

(Sets, packed sets, content package, MPEG-2 picture editing)

Figure 7 illustrates the data structure of the MPEG-2 picture editing metadata.

The transmission order is LSB first to comply with the SDI specification.

The format is now defined in pseudocode representation based on byte transmission order. Values in square brackets indicate the size of the data field in bytes. Note that the data within a byte is ordered in relation to the diagram and does not imply the order of bit stream transmission.

```
MPEG-2_Picture_Editing_Metadata()
{
   picture_edit_bitmap [1]
   picture_coding_parameters [6]
   video_index [15]
   extension_data [var]
}

picture_edit_bitmap ()
{
   edit_flag                2 bits
   error_flag               2 bits
   picture_coding_valid     1 bit
   profile_level_valid      1 bit
   HV_size_valid            1 bit
   user_bitmap_valid        1 bit
}
```

edit_flag (bits b7 b6)
00: No edit
01: Prepicture edit
(the previous element is no longer related to the current element)
10: Postpicture edit
(the following element is no longer related to the current element)
11: Single-frame picture
(neither the previous nor the following elements are related to the current element)

error_flag (bits b5 b4)
00: Error status not known
01: Concealed error
10: Uncorrected error
11: No error

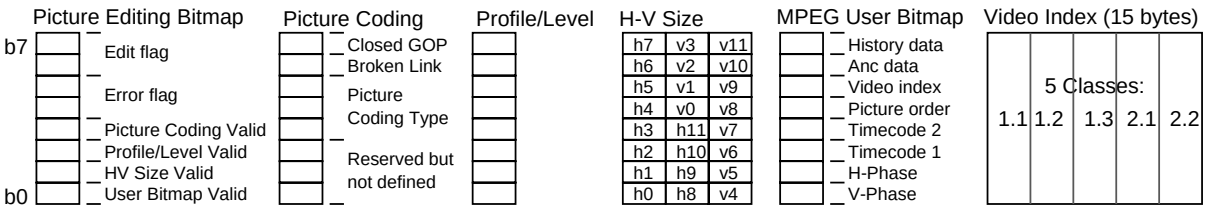

Figure 7 – Illustration of the MPEG-2 picture editing metadata placement

```
if (picture_coding_value == 1)
{
   picture_coding_data ()
   {
      closed_gop                  1 bit (as MPEG)
      broken_link                 1 bit (as MPEG)
      picture_coding_type         3 bits (as MPEG, MSB
                                        in bit b5)
      reserved but not defined    3 bits
   }
}
else
   picture_coding_data [1]:  nonvalid data

if (profile_level_valid == 1)
{
   profile_level       8 bits (as MPEG, MSB in bit b7)

}
else
   profile_level [1]:  nonvalid data

if (HV_size_valid == 1)
{
   HV_size ()
   {
      horizontal_size       12 bits (bits h11 to h0 as
                                defined in figure 7)
      vertical_size         12 bits (bits v11 to v0 as
                                defined in figure 7)
   }
}
else
   HV_size [3]:  nonvalid data

if (user_bitmap_valid == 1)
{
   mpeg_user_data_bitmap ()
   {
      History data:     active = 1, not active = 0
      Ancillary data:   active = 1, not active = 0
      Video index:      active = 1, not active = 0
      Picture order:    active = 1, not active = 0
      Time code 2:      active = 1, not active = 0
      Time code 1:      active = 1, not active = 0
      H-phase:          active = 1, not active = 0
      V-phase:          active = 1, not active = 0
   }
}
else
   mpeg_user_data_bitmap [1]:  nonvalid data
}
```

NOTE – MPEG user data is specified in SMPTE 328M .

```
video_index ()
{
   video_index_class_1.1 [3]
   video_index_class_1.2 [3]
   video_index_class_1.3 [3]
   video_index_class_2.1 [3]
   video_index_class_2.2 [3]


}
```

NOTE – Video_index_class data is defined in SMPTE RP 186.

8.6 AES3 audio editing metadata

Type value = 85_h

Global key value = $0B_h$, 01_h, 01_h, 02_h, 00_h, 00_h, 00_h, 00_h, 00_h

Figure 8 illustrates the data structure of the AES-3 audio editing metadata.

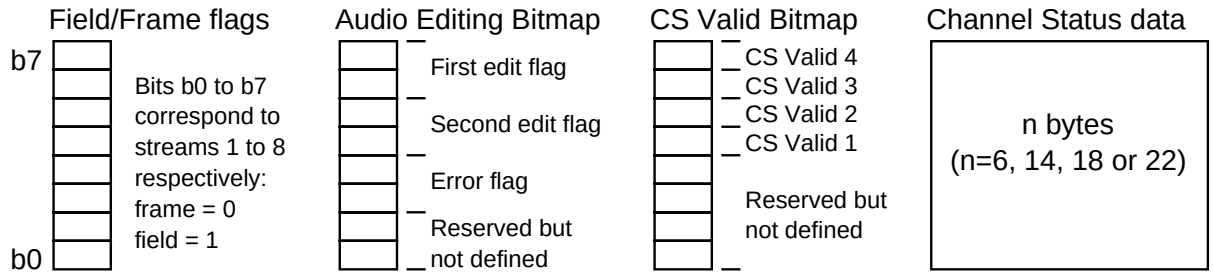

Figure 8 – Illustration of AES3 audio editing metadata placement

The transmission order is LSB first to comply with the SDI specification.

The format is now defined in pseudocode representation based on byte transmission order. Values in square brackets indicate the size of the data field in bytes. Note that the data within a byte are ordered in relation to the diagram and do not imply the order of bit stream transmission.

```
AES3_Audio_Editing_Metadata ()
{
   field_frame_flags [1]

for (ch_number=0; ch_number<8; Ch_number++)
{
   if (stream_valid_flag [ch_number] == 1)
   {
      first_edit_flag                2 bits
      second_edit_flag               2 bits
      error_flag                     2 bits
      reserved but not defined       2 bits
   }
   else
      null_data                      8 bits
}

for (ch_number=0; ch_number<8; Ch_number++)
{
   if (stream_valid_flag [ch_number] == 1)
   {
         cs_valid_4                   1 bit
         cs_valid_3                   1 bit
         cs_valid_2                   1 bit
         cs_valid_1                   1 bit
         reserved but not defined     4 bits

         if (cs_valid_1 == 1)
         {
            for (n=0; n<6; n++)
               cs_byte (n) [1]
         }

         if (cs_valid_2 == 1)
         {
            for (n=6; n<14; n++)
               cs_byte (n) [1]
         }

         if (cs_valid_3 == 1)
         {
            for (n=14; n<18; n++)
               cs_byte (n) [1]
         }

         if (cs_valid_4 == 1)
         {
            for (n=18; n<22; n++)
               cs_byte (n) [1]
         }
      }
   }
}
```

NOTE – When a picture element has two fields, there are two audio edit positions defined by the positions of each picture field. The first and second edit flags apply to the respective first and second picture fields. When a picture is a single frame, only the first edit flag value is valid. Each bit in the field_frame_flags is used to identify whether the corresponding channel is edited on a field or frame basis.

first_edit_flag (bits b7 b6)
00: No edit
01: Prepicture edit (the previous element is no longer related to the current element)
10: Postpicture edit (the following element is no longer related to the current element)
11: Single-frame picture (neither the previous nor the following element is related to the current element).

second_edit_flag (bits b5 b4)
00: No edit
01: Prepicture edit (the previous element is no longer related to the current element)
10: Postpicture edit (the following element is no longer related to the current element)
11: Single-frame picture (neither the previous nor the following element is related to the current element).

error_flag (bits b3 b2)
00: error status not known
01: concealed error
10: uncorrected error
11: no error

NOTE – Stream_valid_flag [ch number] indicates the presence of an audio channel as defined by the stream valid flag bitmap in the 8-channel AES3 element.

8.7 Picture bit stream splicing metadata

Type value = 86_h

Global key value = 04_h, 08_h, 01_h, 02_h, 01_h, 00_h, 00_h, 00_h, 00_h

(Compositional, downstream processing, MPEG, MPEG-2, splicing) The format is defined as follows, following MPEG-2 notation:

```
Picture_Bitstream_Splicing_Metadata ()
{
   In-Point_Present [1]
   Out-Point Present [1]
   Reserved [6]

   If (In_Point_Present)
   {

      In-Point_Splice_Type [4]
      Reserved [3]
      Closed_GOP [1]
   }

   If (Out_Point_Present)
   {
      Out_Point_Splice_Type [4]
      Reserved [4]
   }
}
```

If the in_point_present is 1, the associated picture element shall commence with an in_point as defined by SMPTE 312M with the following exception:

The last sentence of clause 4.3.2.1 does not apply. The effect of this exception is to allow an open GOP at an in-point. The closed_GOP flag shall be used to indicate whether a closed or open GOP exists at the in_point.

If the out_point_present is 1, it indicates that the associated picture element ends with an out_point as defined by SMPTE 312M.

The in_point_splice_type is equal to the value of the splice_type defined in SMPTE 312M and indicates that the video elementary stream meets the constraints corresponding to the value defined in clauses 6.3.2.2 and 6.3.2.3 of SMPTE 312M.

If closed_GOP is 1, it indicates that the in_point meets the constraints defined in SMPTE 312M, clause 6.3.2.1.

The out_point_splice_type is equal to the value of the splice_type as defined in SMPTE 312M, clause 6.2.2, as it pertains to the video elementary stream.

8.8 Decoder buffer delay metadata

Type value = 87_h

Global key value = 01_h, 03_h, $0A_h$, 02_h, 01_h, 00_h, 00_h, 00_h, 00_h

(Essential, image display, delay, MPEG-2, buffer delay)

The decoder buffer delay metadata shall be used only to support the low-latency transfer mode. Its presence in low-latency mode transfer provides an explicit definition of decoder buffer delay. If low-latency mode is used without this metadata, the buffer delay is not defined and a decoder shall set its delay according to its own capabilities.

The decoder buffer delay metadata may be used to support low-latency transfer mode for any picture, audio, or auxiliary data element. The delay shall be specified as a 16-bit word representing the decoder buffer delay as a count of 90-kHz clock periods as follows:

– First byte: Bits b0 to b7 of the 16-bit count word, where bit b0 is the LSB of the first byte.

– Second byte: Bits b8 to b15 of the 16-bit count word, where b8 is the LSB of the second byte.

The 90-kHz clock shall be derived either from the 27-MHz SDTI clock by dividing it by 300, or from the 36-MHz SDTI clock dividing it by 400.

The decoder buffer delay metadata value shall be used to define the period, in 90-kHz clock cycles, between writing the first word of an element into the decoder buffer and reading the first word of that element out of the decoder buffer. The decoder buffer delay value shall be defined as a value large enough to prevent decoder buffer underflow. The maximum decoder buffer requirement to prevent decoder buffer overflow is not specified.

Annex A (informative)
Bibliography

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

SMPTE 305M-1998, Television — Serial Data Transport Interface

Forthcoming SMPTE RP 210, Metadata Dictionary

SMPTE RP 204-2000, SDTI-CP MPEG Decoder Templates