SMPTE ST 2073-7:2022

SMPTE STANDARD

VC-5 Video Essence
Part 7. Metadata

Table of Contents | Page

445 Hamilton Ave., White Plains NY 10601, (914) 761-1100
Approved February 8, 2022

Table of Figures

Table of Tables

Foreword

SMPTE (the Society of Motion Picture and Television Engineers) is an internationally-recognized standards developing organization. Headquartered and incorporated in the United States of America, SMPTE has members in over 80 countries on six continents. SMPTE's Engineering Documents, including Standards, Recommended Practices, and Engineering Guidelines, are prepared by SMPTE's Technology Committees. Participation in these Committees is open to all with a bona fide interest in their work. SMPTE cooperates closely with other standards-developing organizations, including ISO, IEC and ITU.

SMPTE Engineering Documents are drafted in accordance with the rules given in its Standards Operations Manual. This SMPTE Engineering Document was prepared by Technology Committee 10E.

Intellectual Property

At the time of publication no notice had been received by SMPTE claiming patent rights essential to the implementation of this Engineering Document. However, attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. SMPTE shall not be held responsible for identifying any or all such patent rights.

Introduction

This section is entirely informative and does not form an integral part of this Engineering Document.

Metadata is necessary for the proper interpretation of video essence in many applications.

This standard describes how to solve three problems:

(1) Allow an encoder or decoder to access a track of VC-5 essence and a track of metadata that describes the essence,

(2) Represent metadata in a track of VC-5 essence so that metadata is not lost if the track is multiplexed into a different media container, and

(3) Support metadata defined by multiple standards within VC-5 essence, including metadata that does not have a publicly available standard.

Typical media processing frameworks only allow an encoder or decoder to access essence in a single track in a media container, so the encoder or decoder cannot access metadata present in a different track. The method for representing metadata specified in this standard allows encoders and decoders to write or read one or more tracks of metadata that describe the essence.

If metadata is not embedded in the video essence, then the association between video essence and its metadata can be lost. This standard describes how to represent metadata in a VC-5 bitstream so that the association between VC-5 essence and the corresponding metadata survives re-multiplexing the video track into a different media container.

This standard describes a method for representing metadata in VC-5 video essence that includes common formats such as XMP and ALE.

1 Scope

This document extends the VC-5 standard to specify a method for representing metadata in VC-5 essence as the payload of VC-5 chunk elements. Metadata as defined in this standard can include intrinsic metadata that is specific to the VC-5 standard, extrinsic metadata that is defined in other standards, and dark metadata in a non-standard format. This standard defines a method for representing time series data in a VC-5 bitstream.

Intrinsic metadata defined in this standard includes parameters that describe the image represented by VC-5 essence, including the pixel format, color space, and co-siting of subsampled color difference components.

This document specifies how to represent, in the VC-5 bitstream, extrinsic metadata in the following formats: XMP, DPX, SMPTE ST 377-1 MXF Annex F and G essence descriptors, ACES attributes, ALE, and dynamic metadata color volume transforms.

2 Conformance Notation

Normative text is text that describes elements of the design that are indispensable or contains the conformance language keywords: "shall", "should", or "may". Informative text is text that is potentially helpful to the user, but not indispensable, and can be removed, changed, or added editorially without affecting interoperability. Informative text does not contain any conformance keywords.

All text in this document is, by default, normative, except: the Introduction, any section explicitly labeled as "Informative" or individual paragraphs that start with "Note:"

The keywords "shall" and "shall not" indicate requirements strictly to be followed in order to conform to the document and from which no deviation is permitted.

The keywords, "should" and "should not" indicate that, among several possibilities, one is recommended as particularly suitable, without mentioning or excluding others; or that a certain course of action is preferred but not necessarily required; or that (in the negative form) a certain possibility or course of action is deprecated but not prohibited.

The keywords "may" and "need not" indicate courses of action permissible within the limits of the document.

The keyword "reserved" indicates a provision that is not defined at this time, shall not be used, and may be defined in the future. The keyword "forbidden" indicates "reserved" and in addition indicates that the provision will never be defined in the future.

A conformant implementation according to this document 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.

Unless otherwise specified, the order of precedence of the types of normative information in this document shall be as follows: Normative prose shall be the authoritative definition; Tables shall be next; then formal languages; then figures; and then any other language forms.

3 Normative References

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

SMPTE ST 12-1:2014, Time and Control Code

SMPTE ST 258:2004, Transfer of Edit Decision Lists

SMPTE ST 274:2008, Television – 1920 x 1080 Image Sample Structure, Digital Representation and Digital Timing Reference Sequences for Multiple Picture Rates

SMPTE ST 268-1:2014, File Format for Digital Moving-Picture Exchange (DPX)

SMPTE ST 268-2:2018, Digital Moving-Picture Exchange (DPX) – Format Extensions for High Dynamic Range and Wide Color Gamut

SMPTE ST 296:2012, 1280 × 720 Progressive Image 4:2:2 and 4:4:4 Sample Structure – Analog and Digital Representation and Analog Interface

SMPTE ST 298:2009, Universal Labels for Unique Identification of Digital Data

SMPTE ST 314:2005, Television – Data Structure for DV-Based Audio, Data and Compressed Video – 25 and 50 Mb/s

SMPTE ST 377-1:2011, Material Exchange Format (MXF) – File Format Specification

SMPTE ST 2065-1:2012, Academy Color Encoding Specification (ACES)

SMPTE ST 2065-4:2013, ACES Image Container File Layout

SMPTE ST 2073-1:2017, VC-5 Video Essence: Elementary Bitstreams

SMPTE ST 2073-3:2015, VC-5 Video Essence: Image Formats

SMPTE ST 2073-4:2015, VC-5 Video Essence: Subsampled Color Difference Components

SMPTE ST 2073-5:2015, VC-5 Video Essence: Layers

SMPTE ST 2073-6:2015, VC-5 Video Essence: Sections

SMPTE ST 2084:2014, High Dynamic Range Electro-Optical Transfer Function of Mastering Reference Displays

SMPTE ST 2094-2:2017, Dynamic Metadata for Color Volume Transform: KLV Encoding and MXF Mapping

SMPTE ST 2113:2018, Colorimetry of P3 Color Spaces

SMPTE ST 2115:2019, Free Scale Gamut and Free Scale Log Characteristics of Camera Signals

IEEE Std 754-2008, Standard for Floating-Point Arithmetic

ISO/IEC 8601:2004 Data elements and interchange formats – Information interchange – Representation of dates and times

ISO/IEC 9834-8:2014 Information technology – Procedures for the operation of object identifier registration authorities – Part 8: Generation of universally unique identifiers (UUIDs) and their use in object identifiers

ISO/IEC 10646:2017 Information technology – Universal Coded Character Set (UCS)

ISO/IEC 15076-1:2010 Image technology colour management – Architecture, profile format and data structure – Part 1: Based on ICC.1:2010

ISO 16684-1:2012, Graphic technology – Extensible metadata platform (XMP) specification – Part 1: Data model, serialization and core properties

IEC 61834-2:1998, Recording – Helical-scan digital video cassette recording system using 6,35 mm magnetic tape for consumer use (525-60, 625-50, 1125-60 and 1250-50 systems) – Part 2: SD format for 525-60 and 625-50 systems

Recommendation ITU-R BT.601-7 (03/2011) Studio encoding parameters of digital television for standard 4:3 and wide screen 16:9 aspect ratios

Recommendation ITU-R BT.709-6 (06/2015) Parameter values for the HDTV standards for production and international programme exchange

Recommendation ITU-R BT.2020-2 (10/2015) Parameter values for ultra-high definition television systems for production and international programme exchange

Recommendation ITU-R BT.2100-2 (07/2018) Image parameter values for high dynamic range television for use in production and international programme exchange

IETF RFC 3986 Uniform Resource Identifier (URI): Generic Syntax, January 2005
Available from: https://tools.ietf.org/html/rfc3986

IETF RFC 4122 A Universally Unique IDentifier (UUID) URN Namespace
Available from: https://tools.ietf.org/html/rfc4122

NIST Special Publication 330, The International System of Units (SI), 2008 Edition

NIST Special Publication 811, Guide for the Use of the International System of Units (SI), 2008 Edition

W3C Recommendation Extensible Markup Language (XML) 1.0 (Fifth Edition), 26 November 2008
Available from: https://www.w3.org/TR/xml

4 Terms and Definitions

For the purposes of this document, the following terms and definitions shall apply:

4.1 aggregation
set of one or more objects that are contained in a single object

4.2 ALE
text-based metadata file format using comma or tab separated values

4.3 base64
representation of binary data using printable characters as defined in IETF RFC 4648

4.4 big-endian
order of bytes that comprise a number with more significant bytes before less significant bytes

4.5 bitstream position
number of bits counted from the beginning of the bitstream

4.6 class
specification of the implementation of a type

4.7 coordinate
metadata item that comprises one or more numbers that specify location in a coordinate system

4.8 count modulus
maximum integer number of frames within a timecode second

4.9 dark metadata
metadata for which there is no public disclosure of the metadata syntax and semantics

4.10 extrinsic metadata
metadata that is defined by standards that are outside the scope of the VC-5 standard

4.11 fixed-point number
binary representation of a number comprising integer and fractional parts (see section 7.3)

4.12 FourCC
media format identifier consisting of four upper or lower case alphanumeric US-ASCII characters

4.13 frame number
index of an image in a video sequence

4.14 intrinsic metadata
metadata that are defined by the VC-5 standard

4.15 media container
file format that contains one or more media tracks

4.16 media track
one or more sequences of contiguous bytes in a media container that encapsulate a single type of essence

4.17 metadata class
set of metadata items that is defined by a single metadata specification

4.18 metadata chunk
small or large chunk element identified by the chunk tag as containing metadata

4.19 metadata item
smallest unit of metadata information comprising a single metadata value

4.20 metadata object
instance of a metadata class that is the aggregation of one or more metadata items from that metadata class

4.21 metadata tuple
representation of a metadata item

4.22 metadata value
value of a metadata item

4.23 object
instance of a class

4.24 segment
contiguous sequence of 32 bits (four bytes) in a VC-5 bitstream that begins on a segment boundary

4.25 segment boundary
bitstream position that is a multiple of 32 bits (four bytes)

4.26 signed integer
binary representation of an integer using two's complement

4.27 source image
image corresponding to the component arrays output by the image unpacking process

4.28 structured data
data sample that comprises elements that are not all the same data type

4.29 timecode
representation of a frame number as defined in SMPTE ST 12-1

4.30 type
abstract representation of a set of values with common attributes and operations

4.31 unsigned integer
binary representation of an integer that is greater than or equal to 0

4.32 US-ASCII
character set comprising the first 128 characters defined in UTF-8

4.33 UTF-8
multi-byte character encoding as defined in ISO/IEC 10646:2017

4.34 UTF-16

two-byte character encoding as defined in ISO/IEC 10646:2017

4.35 universally unique identifier (UUID)

128-bit number used to identify information in computer systems as defined in ISO/IEC 9834-8:2014

4.36 VC-5 bitstream

contiguous sequence of bytes that represent VC-5 essence

4.37 VC-5 essence

representation of one or more images that conform to the VC-5 Standard for video essence, including any metadata present in the essence by means specified in this standard

4.38 VC-5 standard

SMPTE standards designated ST 2073

5 Overview (Informative)

This standard specifies how to represent metadata in VC-5 essence.

Metadata in a VC-5 bitstream can include intrinsic metadata that is specific to the VC-5 standard, extrinsic metadata that is defined by standards that are outside the scope of the VC-5 standard, and dark metadata that is not described in a published standard.

The syntax of a VC-5 bitstream that contains metadata as defined in this standard is described in section 6. Metadata is organized in a hierarchy of bitstream, metadata chunks, metadata objects, and metadata items (see Figure 1).

The method for representing metadata items in a VC-5 bitstream is specified in section 7.

Different types of metadata can be present in a VC-5 bitstream. As described in section 8, each type of metadata corresponds to a metadata class. The bitstream contains metadata objects that aggregate the metadata items associated with that class instance.

The method for representing metadata objects in the VC-5 bitstream using chunk elements (defined in SMPTE ST 2073-1) is described in section 9.

The decoding process for metadata in a VC-5 bitstream is described in section 10.

A textual representation for metadata objects using XML is described in Annex A. The XML representation can depict how metadata tuples are embedded in a VC-5 bitstream and provides a file format for representing metadata outside of a VC-5 bitstream.

VC-5 intrinsic metadata is described in Annex B. Intrinsic metadata is defined within the VC-5 standards suite and describes the format and characteristics of the source image.

A method for embedding streaming data such as time series in a VC-5 bitstream is described in Annex C. Cameras associate time-varying metadata such as location (GPS coordinates) and calendar date and time with captured images. The method described in Annex C allows arbitrary time series data including location, calendar date and time, and data from sensors such as accelerometers to be embedded in the VC-5 encoded images that comprise a sequence of images.

The method for representing dark metadata in a VC-5 bitstream is described in Annex D. This representation supports vendor-specific metadata. Dark metadata includes any metadata that does not have a published specification.

The remaining annexes specify how to embed external metadata in a VC-5 bitstream. External metadata includes any representation of metadata that is described in a published specification that is not a part of the VC-5 standards suite. The types of external metadata that can be embedded in a VC-5 bitstream as specified in this standard are: contents of an XMP file (Annex E), DPX file header (Annex F), MXF picture essence descriptors (Annex G), ACES attributes (Annex H), contents of an ALE file (Annex I), and dynamic metadata for color volume transforms (Annex J).

6 Bitstream Syntax

A VC-5 bitstream comprises an ordered sequence of tag-value pairs and chunk elements as defined in SMPTE ST 2073-1, with the syntax elements aligned on segment boundaries. Tag-value pairs are one segment comprising a 16-bit tag and 16-bit value. Chunk elements carry data that cannot be represented by a single tag-value pair. The type of data in the payload of a chunk element is identified by the chunk tag.

This standard defines new chunk tags that identify chunk elements that contain metadata. The new chunk elements are called metadata chunks (described in section 9).

The payload of a metadata chunk contains one or more metadata objects. Each metadata object is an instance of a metadata class and comprises an aggregation of metadata items defined by that class (described in section 8).

Metadata objects and items are represented in the bitstream by metadata tuples that comprise the metadata tag, data type, metadata size, repeat count, metadata value, and tuple padding (described in section 7).

7 Metadata Tuples

7.1 Metadata Tuple Syntax

Each metadata tuple shall have the structure defined in Table 1.

Table 1 — Fields that comprise a metadata tuple.

Tuple Field	Byte Position	Size (Bytes)	Description
Metadata Tag	0	4	FourCC that identifies the metadata item (section 7.2)
Data Type	4	1	Single-byte character that specifies the data type of the metadata value (section 7.3)
Metadata Size and Count	5	3	Either a one-byte metadata size followed by a two-byte repeat count or a three-byte size of the metadata value (section 7.4)
Metadata Value	8	Variable	Value of the metadata items (section 7.5)
Tuple Padding	Variable	0-3	Padding to the next segment boundary with bytes having the value zero (section 7.6)

The metadata tuple header shall comprise the metadata tag, data type, metadata size and count as listed in Table 1. The byte order of the metadata header shall be big-endian.

7.2 Metadata Tag

The metadata tag identifies the metadata object or metadata item.

The metadata tag shall be as defined in the tables listed in Table 2.

Table 2 — Tables for metadata tuple tags defined in this document.

Table Description	Table Reference
Generic metadata tags	Table 3
Metadata tags for metadata objects	Table 5
Metadata tags for intrinsic metadata	Table B.1
Device information for time-series data	Table C.1
Stream information for time-series data	Table C.2
Dark metadata	Table D.1
Metadata items for XMP metadata objects	Table E.1
Metadata items for DPX metadata objects	Table F.1
Metadata items for MXF metadata objects	Table G.1
Metadata items for ACES attributes	Table H.1
Metadata items for ALE metadata objects	Table I.1
Metadata items dynamic metadata color volume transform (DMCVT)	Table J.1

Table 3 — List of generic metadata tags and data types for metadata tuples.

Metadata Tag	Data Type	Description
'FREE'	'B'	Metadata tuple representing free space in the payload of the metadata chunk element

NOTE: The metadata tags defined in Table 3 are common to all metadata classes.

7.3 Data Types

The data type code identifies the data type of the metadata value in a metadata tuple.

The data type code shall be one of the single-byte codes in US-ASCII as listed in Table 4.

Whether the metadata size and count field (section 7.4) includes a repeat count shall be as indicated by the repeat count column in Table 4 for each data type code.

The character string data types (data type codes 'c', 'u', and 'w') shall use the character encodings defined in ISO/IEC 10646.

Numerical data types shall be as specified in Table 4. Integer data types are numerical data types that are not floating-point (data type code 'h', 'f' or 'd') or rational (data type code 'r' or 'R'). The integer data types shall be as specified in Table 4.

The IEEE floating-point data types (data type codes 'h', 'f', and 'd') shall be as defined in IEEE Std 754.

The fixed-point number formats shall be one of the following:

> Q15.16 is the concatenation of a 16-bit signed integer A immediately followed by a 16-bit unsigned integer B representing the number A.B with integer part A and fractional part B
>
> Q31.32 is the concatenation of a 32-bit signed integer A immediately followed by a 32-bit unsigned integer B representing the number A.B with integer part A and fractional part B

Boolean metadata values shall be represented by an 8-bit unsigned integer (data type code ‘B’). The value 0 shall indicate false, all other values shall indicate true. The value should be 0 or 1.

The character encoding of XML (data type code ‘x’) shall be as specified in W3C Recommendation Extensible Markup Language (XML).

Blocks of binary data (blobs) should be represented using the blob data type (data type code ‘X’).

NOTE 1: The use of arrays of arrays (described in section 7.8) to represent blocks of binary data is deprecated.

Table 4 — Data types for metadata items.

Data Type Code	Data Type Description	Repeat Count	Numerical Data Type	Integer Data Type
0x0	Nested metadata tuples	No	No	No
‘c’	US-ASCII string without trailing nul characters	No	No	No
‘b’	Signed 8-bit integer	Yes	Yes	Yes
‘B’	Unsigned 8-bit integer or a byte	Yes	Yes	Yes
‘h’	IEEE floating-point value (half precision)	Yes	Yes	No
‘f’	IEEE floating-point value (single precision)	Yes	Yes	No
‘d’	IEEE floating-point value (double precision)	Yes	Yes	No
‘E’	Metadata object	No	No	No
‘F’	Four character code (FourCC)	Yes	No	No
‘G’	A 128-bit Universally Unique Identifier (UUID) as defined in IETF RFC 4122	Yes	No	No
‘l’	Signed long integer (32 bits)	Yes	Yes	Yes
‘L’	Unsigned long integer (32 bits)	Yes	Yes	Yes
‘j’	Signed long long integer (64 bits)	Yes	Yes	Yes
‘J’	Unsigned long long integer (64 bits)	Yes	Yes	Yes
‘P’	Metadata value that comprises tuples representing parameters	No	No	No
'q'	Fixed-point number in the format Q15.16	Yes	Yes	No
'Q'	Fixed-point number in the format Q31.32	Yes	Yes	No

Data Type Code	Data Type Description	Repeat Count	Numerical Data Type	Integer Data Type
‘r’	Ratio comprising a signed 16-bit integer numerator and an unsigned 16-bit integer denominator in that order	Yes	Yes	No
‘R’	Ratio comprising two unsigned 16-bit integers in the order numerator followed by denominator	Yes	Yes	No
‘s’	Signed short integer (16 bits)	Yes	Yes	Yes
‘S’	Unsigned short integer (16 bits)	Yes	Yes	Yes
‘x’	XML represented as a character string	No	No	No
‘u’	Multi-byte character string (UTF-8)	No	No	No
‘w’	Wide character string (UTF-16)	No	No	No
‘U’	Fixed-length 16-byte SMPTE Universal Label (UL) as defined in SMPTE ST 298	Yes	No	No
‘X’	Block of binary data (blob)	No	No	No

The data type code 0 means that the metadata value comprises nested metadata tuples (section 7.7).

NOTE 2: Data types with metadata values that can be larger than 255 bytes cannot have a repeat count because the object size is only one byte when the size and count field contains a repeat count.

NOTE 3: Some metadata items are numbers and require a numerical or integer data type.

7.4 Metadata Size and Count

7.4.1 Object Size

If the data type of the metadata tuple has a repeat count as listed in Table 4, then

- The metadata size and count field shall comprise a one-byte metadata size (section 7.4.2) followed by a two-byte repeat count (section 7.4.3) and
- The length of the metadata value shall be the product of the metadata size times the repeat count.

If the data type of the metadata tuple does not have a repeat count, then

- The metadata size and count field shall be a three-byte field that specifies the length of the metadata value.

7.4.2 Metadata Size

If the data type of the metadata tuple has a repeat count as listed in Table 4, then the single byte metadata size starting at byte position 5 shall be the size of each data element in the metadata value (in bytes). The number of data elements is determined by the repeat count.

7.4.3 Repeat Count

If the data type of the metadata tuple has a repeat count as listed in Table 4, then the repeat count shall be the unsigned integer comprising the 2 bytes starting at byte position 6 in the metadata tuple. The repeat count shall be the number of data elements in the metadata value. The value of the repeat count shall be a 16-bit unsigned integer in the range 1 to 65535.

7.5 Metadata Value

The size of the metadata value does not include the metadata tuple header and does not include tuple padding.

7.6 Tuple Padding

The metadata tuple shall be padded with 0 to 3 bytes with value zero to fill the last segment in the metadata tuple to the next segment boundary.

7.7 Nested Metadata Tuples

The metadata value of a nested metadata tuple (data type code 0) shall comprise zero or more metadata tuples and nothing more.

7.8 Arrays of Arrays (Informative)

The size and repeat count described in section 7.4 can be used to represent an array of vector-valued metadata. The size of each vector in bytes is represented by the tuple metadata size and the number of vectors in the array is represented by the tuple repeat count.

EXAMPLE: An accelerometer produces a series of measurements. Each measurement is a 3-vector and the size of each vector equals 3 times the size of each element in the vector (in bytes). The repeat count is the number of measurements in the series. Measurement vectors are contiguous in the bitstream. The product of size times count equals the total size of the metadata value (in bytes).

8 Metadata Classes

8.1 Data Model

Metadata are organized into metadata classes. Each class is associated with a specification of intrinsic metadata (section 8.3), extrinsic metadata (section 8.4), or dark metadata (section 8.5).

A metadata object is an instance of a metadata class. The metadata value of each metadata object shall comprise one or more metadata items defined in its metadata class and nothing more.

If the bitstream contains more than one image section (defined in SMPTE ST 2073-6), then each image section shall contain at most one metadata object from each metadata class; otherwise the entire bitstream shall contain at most one metadata object from each metadata class.

A metadata object shall aggregate at most one instance of each metadata item defined in that class.

A metadata object shall be represented in the bitstream as a metadata tuple. Each metadata item in a metadata object shall be represented in the bitstream by exactly one metadata tuple. Aggregation of metadata items in metadata objects shall be represented using nested tuples (see Figure 1).

8.2 Metadata Class Tags and Data Types

The metadata tag and data type for a metadata object shall be as listed in Table 5.

Table 5 — Metadata tags and data types for metadata objects.

Metadata Tag	Data Type	Description
‘CFHD’	‘E’	VC-5 intrinsic metadata defined in 0
‘GPMF’	‘E’	Time series data defined in 0
‘XMPD’	‘E’	XMP metadata as defined in ISO 16684-1:2012 and described in Annex E
‘DPXF’	‘E’	DPX file header as defined in SMPTE ST 268-1 and SMPTE ST 268-2 and described in Annex F
‘MXFD’	‘E’	MXF Annex F and G Essence Descriptors as defined in SMPTE ST 377-1 and described in Annex G
‘ACES’	‘E’	ACES attributes as defined in SMPTE ST 2065-4 and described in Annex H
‘ALEM’	‘E’	Metadata from an ALE file described in Annex I
‘DMCT’	‘E’	Dynamic metadata color volume transform (DMCVT) as defined in SMPTE ST 2094-2 and described in Annex J
‘DARK’	‘E’	Dark metadata defined in 0

8.3 Intrinsic Metadata

The byte order of the metadata tuples aggregated in intrinsic metadata objects (metadata tag ‘CFHD’) and time series data metadata objects (metadata tag ‘GPMF’) shall be big-endian.

8.4 Extrinsic Metadata

The metadata value of an extrinsic metadata object shall be as specified in the corresponding metadata standard (see the appropriate annex for details).

8.5 Dark Metadata

The dark metadata class defines three metadata items:

1. Dark metadata identification code,
2. Dark metadata identification string,
3. Dark metadata item.

A dark metadata object shall be an instance of the dark metadata class.

The dark metadata object shall aggregate:

A dark metadata identification code, a dark metadata identification string, or both; and

A block of dark metadata.

9 Metadata Chunks

The bitstream can contain zero or more metadata chunks. A metadata chunk shall be a small chunk element or a large chunk element. The chunk tags for small and large metadata chunks shall be as listed in Table 6.

Table 6 — Tag numbers for metadata small and large chunk elements.

Chunk Tag	Tag Number	Description
SmallMetadataTag	0x4010	Tag number for a small metadata chunk element
LargeMetadataTag	0x61	Tag number for a large metadata chunk element

The payload of a metadata chunk shall comprise one or more metadata tuples and nothing more.

Metadata tuples shall not be split across multiple chunks.

All metadata chunks associated with an image represented in the bitstream shall occur in the bitstream before the last subband of that image.

If multiple images are represented in the bitstream using image sections, as defined in SMPTE ST 2073-6, then the metadata chunks in each image section shall only apply to the image represented in that image section.

Default values shall be used for metadata items that are not present in an image section.

A metadata chunk element and its nested payload of metadata objects and metadata items is diagrammed in Figure 1.

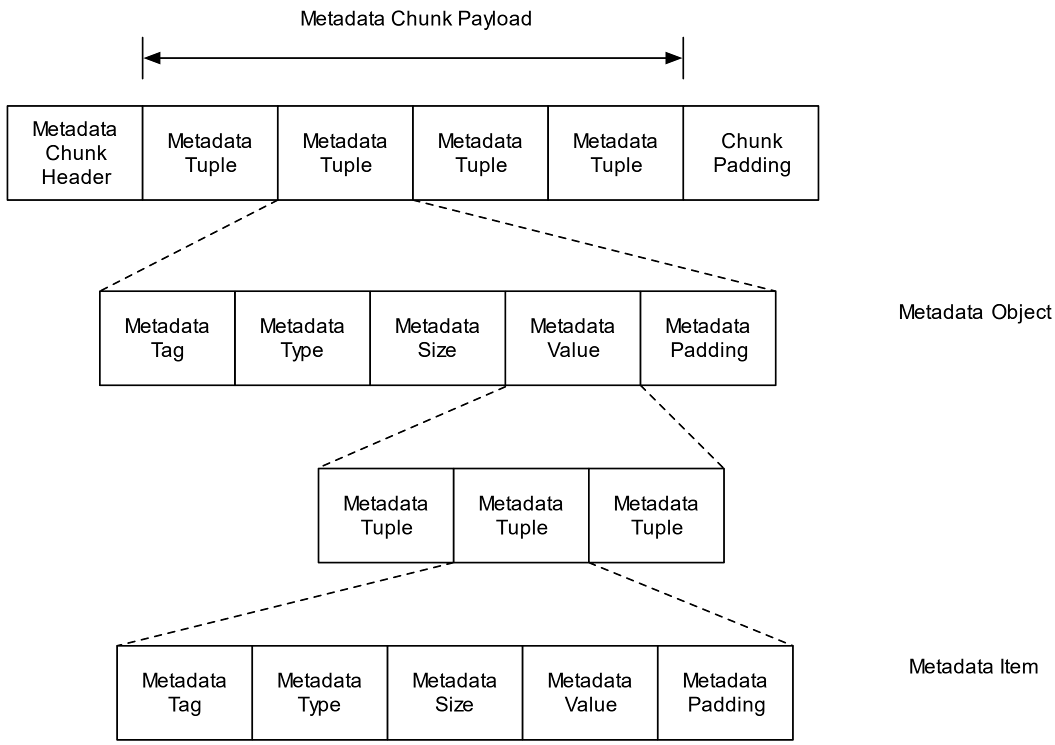

Figure 1 — Metadata chunk element and the nested structure of metadata objects and metadata items.

10 Decoding Process

The decoder shall parse the metadata tuples from all metadata chunks in the same order as the metadata tuples occur in the bitstream. If an image section (or the entire bitstream if image sections are not used), contains more than one metadata tuple with the same metadata tag, then only the last metadata tuple instance encountered in the image section (or bitstream) shall be retained, unless any of the following conditions are true:

1. The metadata item is layer metadata, in which case multiple layer metadata items are allowed, but only the last layer metadata item for a given layer number shall be retained,

2. The metadata class is streaming metadata (metadata class ‘GPMF’), in which case multiple metadata items with the same tag are allowed.

The decoding process shall preserve the byte order of every metadata tuple.

If a metadata item is not present in the bitstream, then the default value of the metadata item shall be used.

The decoding process can accumulate a database of metadata items for each image section (or for the entire bitstream if image sections are not used) by parsing the metadata tuples in metadata chunk elements. The manner in which an application makes use of the metadata database is out of scope.

Annex A Metadata XML Representation (Informative)

A.1 Metadata XML Syntax

This annex describes one method for mapping metadata in a VC-5 bitstream into XML.

The instances of the metadata formats defined in this document can be represented as XML to provide a textual representation of the metadata. The syntax of XML is defined in W3C Recommendation Extensible Markup Language (XML).The namespace prefix for this representation is `vc5`.

Each metadata tuple is represented as an XML element with the element tag `vc5:tuple`. The metadata tag is represented by the tag attribute of the XML element with the attribute value equal to the FourCC. The metadata type is represented by the type attribute of the XML element with the attribute value equal to the single character that represents the data type as listed in Table 4.

The size of each data element in the metadata value is represented by the size attribute of the XML element with the attribute value equal to the size in bytes. If the size attribute is not present in the XML element or has the value 0, then the size of each data element in the metadata value can be calculated from the metadata type in that XML element, but only if the element is a scalar value. It is good practice to always include the size attribute.

If the metadata tuple has a repeat count, then the repeat count is represented as the count attribute of the XML element with the attribute value being the number of elements in the metadata value. Refer to the discussion of arrays of arrays in section 7.8.

The metadata tuple padding can be represented by the padding attribute of the XML element with attribute value equal to the number of bytes of padding. If the padding attribute is not present in the XML element, then the metadata tuple padding is computed from the metadata size and count (as specified in section 7.4) as the minimum number of bytes required to pad the metadata tuple to a segment boundary.

The metadata value is represented by one of the following methods:

1. Values without a repeat count can be represented using the value attribute,
2. Values without a repeat count can be represented in the body of the element,
3. Values with a repeat count can be represented as a list in the body of the element with the list elements separated by whitespace,
4. A block of binary data in the form of an array of arrays of unsigned 8-bit integers (data type ‘B’) can be represented as a base64 encoded metadata value (deprecated).
5. A block of binary data can be represented using data type code ‘X’ with base64 encoding (preferred).

A.2 Intrinsic Metadata Item

If the `count` attribute is not present in the XML element and the `tag` attribute is "CFHD", then the metadata value is represented by the `value` attribute or the content of the XML element. The `value` attribute is preferred for small metadata values.

An example of using XML to represent an intrinsic metadata item that specifies that the color space is BT.709 (see Annex B.7) is as follows:

```
<vc5:tuple tag="CLSY" type="B" size="1" value="6" padding="3" />
```

The following example shows how the descriptors for the layers in a VC-5 Part 5 bitstream containing a stereo pair are represented using XML:

```
<vc5:tuple tag="LAYR">
    <vc5:tuple tag="LAYN" type="S" value="1" />
    <vc5:tuple tag="LAYD" type="c">Stereo left image</vc5:tuple>
</vc5:tuple>
<vc5:tuple tag="LAYR">
    <vc5:tuple tag="LAYN" type="S" value="2" />
    <vc5:tuple tag="LAYD" type="c">Stereo right image</vc5:tuple>
</vc5:tuple>
```

A.3 Streaming Metadata Item

If the count attribute is present in the tuple, then the metadata value is the content of the XML element and comprises one or more value elements in the `vc5` namespace with one value element per data sample.

An example of using XML to represent a streaming metadata item that specifies samples from a time series (see Annex C) is as follows:

```
<vc5:tuple tag="DEVC" type="0" size="28">
    <vc5:tuple tag=" STRM" type="0" size="20">
        <vc5:tuple    tag="GYRO"    type="f"    size="4"    count="3">1.45    2.30
                    1.34</vc5:tuple>
    </vc5:tuple>
</vc5:tuple>
```

A.4 Intrinsic Metadata Object

If the `tag` attribute is "CFHD" or "GPMF" and the `type` attribute is "E", then the metadata object contains intrinsic metadata. The metadata value is the content of the XML element and comprises one or more XML elements that represent nested metadata tuples.

An example of using XML to represent an intrinsic metadata object that aggregates intrinsic metadata items (see Annex B) is as follows:

```
<vc5:tuple tag="CFHD" type="E">
    <vc5:tuple tag="CLSY" type="B" size="1" value="13" padding="3" />
    <vc5:tuple tag="GAMA" type="P">
        <vc5:tuple tag="GAMp" type="f" size="4" value="1.2" />
    </vc5:tuple>
    <vc5:tuple tag="RATE" type="L" size="4" value="30000" />
    <vc5:tuple tag="SCAL" type="L" size="4" value="1001" />
</vc5:tuple>
```

A.5 Extrinsic Metadata Object

If the `type` attribute is “E” and the `tag` attribute is not “CFHD” or “GPMF” or “DARK”, then the metadata object contains extrinsic metadata and the metadata value is the content of the XML element as specified in the annex for the type of extrinsic metadata as determined by the metadata tag (section 8.2).

An example of using XML to represent an extrinsic metadata object is as follows:

```
<vc5:tuple tag="XMPD" type="E">
    <vc5:tuple tag="XMPd" type="x">
        <![CDATA[<?xpacket begin="" id="W5M0MpCehiHzreSzNTczkc9d"?>
          <x:xmpmeta
              xmlns:x="adobe:ns:meta/"
              x:xmptk="Adobe XMP Core 5.4-c002 1.000000, 0000/00/00-00:00:00     ">
            <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
                <rdf:Description
                      rdf:about=""
                      xmlns:exif="http://ns.adobe.com/exif/1.0/">
                    <exif:PixelXDimension>912</exif:PixelXDimension>
                    <exif:PixelYDimension>687</exif:PixelYDimension>
                    <exif:ExifVersion>0220</exif:ExifVersion>
                </rdf:Description>
            </rdf:RDF>
          </x:xmpmeta>
        <?xpacket end="w"?>]]>
    </vc5:tuple>
</vc5:tuple>
```

XMP metadata can be embedded in the XML representation of VC-5 metadata using a CDATA element to distinguish the XML representation of XMP from the XML that represents the VC-5 metadata.

A.6 Dark Metadata Object

If the `type` attribute is “E” and the `tag` attribute is “DARK”, then the metadata object contains dark metadata.

An example of representing dark metadata encoded as base64 (IETF RFC 4648) using XML is as follows:

```
<vc5:tuple tag="DARK" type="E">
    <vc5:tuple tag="VENI" type="G" size="32"
        value="63041a95-c636-454b-810b-89ceb899445e" />
    <vc5:tuple tag="VENS" type="c" size="10" value="Vendor XYZ" />
    <vc5:tuple tag="VEND" type="B" size="51" padding="1">
        QW4gZXhhbXBsZSBvZiByZXByZXNlbnRpbmcgZGFyayBtZXRhZGF0YSB1c2luZyBYTUwNCg==
    </vc5:tuple>
</vc5:tuple>
```


A.7 Metadata Chunk Element

A metadata chunk can be represented as an XML element using a `vc5:chunk` element. The chunk tag is represented using a `tag` attribute, the size is represented using a `size` attribute, and padding is represented using a `padding` attribute. The size of a chunk is in units of segments and the amount of padding is the number of zero bytes following the chunk payload.

An example of using XML to represent a chunk of metadata is provided as follows:

```
<vc5:chunk tag="0x4010" size="10" padding="0">
   <vc5:tuple tag="CFAP" type="c" value="RGGB"/>
   <vc5:tuple tag="CLSY" type="B" value="6" />
   <vc5:tuple tag="LOGC" type="P">
      <vc5:tuple tag="LOGb" type="f" value="1.8" />
   </vc5:tuple>
</vc5:chunk>
```

The chunk tag indicates whether the metadata chunk is a small or large chunk element.

Annex B VC-5 Intrinsic Metadata (Normative)

B.1 Intrinsic Metadata Overview

VC-5 image metadata is intrinsic metadata in the VC-5 bitstream as descriptive annotation of the image represented by the VC-5 essence.

B.2 Intrinsic Metadata Tags and Data Types

The metadata tuples tags and data types for intrinsic metadata shall be as defined in Table B.1.

Table B.1. VC-5 intrinsic metadata tags and data types.

Metadata Tag	Data Type	Description
‘ROWI’	‘S’	Source image row offset (section B.3)
‘COLI’	‘S’	Source image column offset (section B.3)
‘NCOL’	‘S’	Source image width (section B.3)
‘NROW’	‘S’	Source image height (section B.3)
‘PFMT’	‘B’	Source image pixel format (section B.4)
‘ALPH’	‘L’	Source image constant alpha component value (section B.5)
‘ALPM’	‘B’	Source image alpha pre-multiplied flag (section B.5)
‘CFAP’	‘c’	Color filter array pattern (section B.6)
‘CLSY’	‘B’	Color space (section B.7)
‘CMIN’	‘L’	Minimum value of the RGB color components (section B.8)
‘CMAX’	‘L’	Maximum value of the RGB color components (section B.8)
‘AMIN’	‘L’	Minimum value of the alpha components (section B.8)
‘AMAX’	‘L’	Maximum value of the alpha components (section B.8)
‘BLKR’	‘L’	Black reference level (section B.8)
‘WHTR’	‘L’	White reference level (section B.8)
‘COLR’	‘L’	Color range (section B.8)
‘CDCS’	‘B’	Color difference component co-siting (section B.10)
‘RATE’	‘J’	Number of clock ticks per second (section B.11)
‘SCAL’	‘J’	Number of clock ticks per frame (section B.11)
‘FRMN’	‘L’	Frame number corresponding to the image represented in the bitstream (section B.12)
‘TIMB’	‘L’	Count modulus for a video sequence (section B.12)
‘TIMD’	‘B’	Drop frame flag (section B.12)
‘TIMS’	‘c’	Timecode corresponding to the frame number origin (section B.12)
‘FRMZ’	‘L’	Frame number origin (section B.12)
‘LAYR’	‘P’	Layer descriptive metadata (section B.13)
‘ICCP’	‘B’	International Color Consortium (ICC) color profile (section B.14)

B.3 Source Image Dimensions

The metadata item for the source image width (metadata tag ‘SIWD’) specifies the number of samples in each row of the sample array corresponding to the source image. The metadata item for the source image height (metadata tag ‘SIHT’) specifies the number of rows in the sample array corresponding to the source image. See SMPTE ST 2073-3, section 7, for information about sample arrays.

For color difference component subsampling, the source image dimensions shall be the number of rows and columns in the luminance component array as defined in SMPTE ST 2073-4, section 8.2.

The source image row offset (metadata tag ‘ROWI’) is the index of the row in the source image corresponding to the lowest numbered row index in the image represented in the bitstream.

The source image column offset (metadata tag ‘COLI’) is the index of the column in the source image corresponding to the lowest numbered column index in the image represented in the bitstream.

Source image dimensions shall use an integer data type as defined in section 7.3.

NOTE: The VC-5 bitstream includes the dimensions of the encoded image (SMPTE ST 2073-3, section 7.4, and SMPTE ST 2073-4, section 8.2). The source image dimensions and offset provide information that can allow the image repacking process to position the decoded image within a larger raster or output a sub-image of the decoded image.

B.4 Source Image Pixel Format

The metadata item for the source image pixel format (metadata tag ‘PFMT’) specifies the pixel format of the source image.

The metadata value of a source image pixel format metadata tuple shall be an RGBALayout data item as defined in SMPTE ST 377-1, section G.2.40.

NOTE: The metadata size is an RGBALayout data item is 16 bytes.

B.5 Source Image Constant Alpha

If this metadata item is present, then the alpha value is a constant value, the alpha component arrays shall not be present in the bitstream, and the alpha value shall be the value of this metadata item.

The metadata item for the source image constant alpha component value (metadata tag ‘ALPH’) specifies the alpha component value when the alpha component array is not present in the bitstream.

The source image constant alpha component value shall use an integer data type as defined in section 7.3.

The data type of the source image alpha pre-multiplied flag (metadata tag ‘ALPM’) shall be a Boolean data type as defined in section 7.3.

If present, the source image alpha pre-multiplied flag (metadata tag ‘ALPM’) indicates whether the source image constant alpha component value is pre-multiplied. If the source image alpha pre-multiplied flag is true, then the source image alpha component array shall be pre-multiplied, otherwise the source image alpha component array shall not be pre-multiplied.

If the source image alpha pre-multiplied flag is not present, then the source image constant alpha component value shall not be pre-multiplied.

B.6 Color Filter Array Pattern

The metadata item for the color filter array pattern (metadata tag ‘CFAP’) specifies the order and type of the components in each pattern element in the image represented in the bitstream as defined in SMPTE ST 2073-3:2015, Section 7.3.

NOTE 1: This metadata item represents the color components in the pattern elements of the sample array corresponding to the source image, prior to the component transform and component permutation applied by the encoder as described in SMPTE ST 2073-3.

The metadata value of a color filter array pattern metadata tuple shall be a contiguous sequence of US-ASCII characters in the same order as the components of each pattern element with the values listed in Table B.2.

Table B.2 — Values that designate the type of the components in a pattern element.

Value	Description
‘R’	Red component
‘G’	Green component
‘B’	Blue component

NOTE 2: The characters for the component types are defined in SMPTE 377-1:2011, Section G.2.40.

B.7 Color Space

The color space metadata tuple (metadata tag ‘CLSY’) specifies the color space of the source image including the color primaries, white point, transfer characteristic, and coding equation.

The color primaries, white point and coding equation of the image represented in the bitstream are the same as those specified by the color space metadata tuple. The transfer characteristic can be different if the encoding curve metadata tuple is present in the bitstream as described in section B.9.

The metadata value in a color space metadata tuple shall be one of the values listed in Table B.3.

Table B.3 — Valid values for the color space metadata item.

Metadata Value	Description
0	Default color space as defined in SMPTE ST 2073-3:2015, Annex C
1	Unspecified linear color space
2-5	Reserved
6	Color space as defined in ITU-R BT.709
7	Color space for 625 line systems as defined in ITU-R BT.601
8	Color space for 525 line systems as defined in ITU-R BT.601
9-12	Reserved
13	SMPTE ST 2065-1:2012 Academy Color Encoding Specification (ACES)
14	Color space as defined in ITU-R BT.2020-2 non-constant luminance per footnote (2) to Table 4 in that standard

Metadata Value	Description
15	P3D65 as defined in SMPTE ST 2113 with the transfer function as specified in SMPTE ST 2084
16	P3DCI as defined in SMPTE RP 431-2, Table A.1
17-20	Reserved
21	Perceptual Quantization (PQ) system reference non-linear transfer functions with R'G'B' or non-constant luminance $Y'C'_BC'_R$ signal format as defined in Recommendation ITU-R BT.2100
22	Perceptual Quantization (PQ) system reference non-linear transfer functions and constant intensity IC_TC_P signal format as defined in Recommendation ITU-R BT.2100
23	Hybrid-Log Gamma (HLG) system reference non-linear transfer functions and non-constant luminance $Y'C'_BC'_R$ signal format as defined in Recommendation ITU-R BT.2100
24	Hybrid-Log Gamma (HLG) system reference non-linear transfer functions and constant intensity IC_TC_P signal format as defined in Recommendation ITU-R BT.2100

If a color space metadata tuple is not present in the metadata chunk, then the color space shall be as listed for metadata value 0 in Table B.3.

NOTE: The color space metadata tuple is intended to provide guidance to the decoder so that the image represented in the bitstream can be reconstructed to the original color space of the source image.

B.8 Color Range

The color range metadata items specify the range of component values in the source image.

NOTE: This color range metadata items are intended to distinguish between video range and full range (computer range) pixel representations.

The metadata value of the color minimum metadata tuple (metadata tag ‘CMIN’) shall be the smallest value of RGB components that occur in the source image.

The metadata value of the color maximum metadata tuple (metadata tag ‘CMAX’) shall be the largest value of RGB components that occur in the source image.

If the RGB component minimum metadata tuple is not present in the metadata chunk, then the default value shall be 0.

If the RGB component maximum metadata tuple is not present in the metadata chunk, the default value shall be the maximum value of the color component as determined by the precision of the color component values represented by the value of the **BitsPerComponent** tag-value pair.

The **BitsPerComponent** tag-value pair is defined in SMPTE ST 2073-1.

The metadata value of the alpha minimum metadata type (metadata tag ‘AMIN’) shall be the smallest value of the alpha components that occur in the source image.

The metadata value of the alpha maximum metadata type (metadata tag ‘AMAX’) shall be the largest value of the alpha components that occur in the source image.

If the alpha minimum metadata tuple is not present in the metadata chunk, then the default value shall be 0.

If the alpha maximum metadata tuple is not present in the metadata chunk, the default value shall be the maximum value of the color component as determined by the precision of the color component values represented by the value of the **BitsPerComponent** tag-value pair.

The metadata value of the black reference level (metadata tag ‘BLKR’) shall be the value of black for the luminance component in the color difference component source image.

The metadata value of the white reference level (metadata tag ‘WHTR’) shall be the value of white for the luminance component in the color difference component source image.

If the black reference level metadata tuple is not present in the metadata chunk, then the default value shall be 0.

If the white reference level metadata tuple is not present in the metadata chunk, then the default value shall be the maximum value of the color component as determined by the precision of the color component values represented by the value of the **BitsPerComponent** tag-value pair.

The metadata value of the color range metadata tuple (metadata tag ‘COLR’) shall be the range of color difference components in the source image.

If the color range metadata tuple is not present in the metadata chunk, then the default value shall be the precision of the color component values represented by the value of the **BitsPerComponent** tag-value pair.

Metadata tuples for the RGB component ranges (metadata tags ‘CMIN’ and ‘CMAX’), alpha component ranges (metadata tags ‘AMIN’ and ‘AMAX’), black and white reference levels (metadata tags ‘BLKR’ and ‘WHTR’), and the range of color difference components (metadata tag ‘COLR’) shall use an integer data type as defined in section 7.3.

B.9 Encoding Curve

B.9.1 Encoding Curve Metadata

If and only if the encoding curve metadata tuple is present in the bitstream, then the following shall apply:

1. The color space transfer function of the image represented in the bitstream is specified by the encoding curve metadata tuple.

2. The metadata tag of the encoding curve metadata tuple specifies the forward encoding curve formula.

3. The metadata tag of the encoding curve metadata tuple is one of the tags listed in Table B.4.

NOTE 1: The inverse encoding curve is the transfer function that a decoder can apply to the decoded component arrays to convert them to linear light prior to applying an application-specific encoding curve.

NOTE 2: The encoding curve is not usually the transfer function for the color space of the source image.

The metadata value of an encoding curve metadata tuple shall comprise zero or more metadata tuples that specify the parameters of the encoding curve formula, and nothing more. The nested metadata tuples that represent encoding curve parameters are called parameter tuples to distinguish them from other tuples that represent metadata items and objects.

Each of the parameter tuples shall use one of the numerical data types defined in section 7.3.

Unless otherwise specified:

> The input value x is proportional to the light intensity for that component with input value 1 corresponding to the reference white value,
>
> The output value y is represented in the bitstream as specified in SMPTE ST 2073-1 with the output value 1 corresponding to the reference white value.

Table B.4 — Four character codes that specify the encoding curve formula.

Metadata Tag	Description
‘LOGA’	Log encoding curve (section B.9.2)
‘GAMA’	Gamma encoding curve (section B.9.3)
‘LINR’	Linear encoding curve (section B.9.4)
‘FSLG’	FS-Log encoding curve (section B.9.5)
‘LOGC’	Log C encoding curve (section B.9.6)
‘PQEC’	Perceptual quantizer encoding curve (section B.9.7)
‘HLGE’	Hybrid log-gamma encoding curve (section B.9.8)

B.9.2 Log Encoding Curve

The log encoding curve shall use the following formula to map input values x to output values y:

$$\mathrm{y} = \log_{10}(\mathrm{x}\,(\mathrm{b} - 1.0) + 1.0)/\log_{10}(\mathrm{b})$$

where parameter b is the metadata value of a parameter tuple with metadata tag ‘LOGb’.

NOTE: Color correction is performed in log space per ASC recommendations.

B.9.3 Gamma Encoding Curve

The gamma encoding curve shall use the following formula to map input values x to output values y:

$$\mathrm{y} = \mathrm{x}^{(1/\gamma)}$$

where parameter γ is the metadata value of a parameter tuple with metadata tag ‘GAMp’.

B.9.4 Linear Encoding Curve

The linear encoding curve specifies that the color components were converted to linear space and the input to the encoding process is in linear space. The size of the metadata value shall be zero.

B.9.5 Free Scale Log Encoding Curve

The free scale log (FS-Log) encoding curves shall be as specified in SMPTE ST 2115:2019, section 7.2.

The metadata value of the encoding curve metadata tuple shall comprise a single parameter tuple with metadata tag ‘FSCL’ that specifies the specific encoding curve by number as listed in Table B.5.

Table B.5 — Free scale log (FS-Log) encoding curves.

Encoding Curve Number	Description	Section Reference in SMPTE ST 2115:2019
13	Camera Log S3	7.2.1
20	Camera Log V	7.2.2
32	Camera Log C2	7.2.3
33	Camera Log C3	7.2.4

The metadata value for the encoding curve number shall be an integer data type as listed in Table 4.

B.9.6 Parameterized Log Encoding Curve

The parameterized log encoding curve shall use the following formula to map input values x to output values y:

If $x > t$, then $y = c \log_{10}(a\, x + b) + d$

otherwise $y = e\, x + f$

where parameters t, a, b, c, d, e, and f are the values of parameter tuples with metadata tags ‘LOGt’, ‘LOGa’, ‘LOGb’, LOGc’, LOGd’, ‘LOGe’, and ‘LOGf’, respectively, in the metadata value of the log C encoding curve metadata tuple. The curve is described in SMPTE RDD 31:2014.

The input values are relative exposure as defined in RDD 31:2014, repeated here:

From a given linear 16-bit unsigned integer wide-gamut RGB V_{sens}, the corresponding floating-point relative exposure value V_{ev} is

$$V_{ev} = \left(\frac{V_{sens} - 256.0}{65535.0}\right)\frac{0.18\, EI}{4.0}$$

with V_{sens} being converted to floating-point prior to black subtraction, and with EI similarly being converted to floating-point prior to its use as a scaling factor.

B.9.7 Perceptual Quantizer Encoding Curve

The Perceptual Quantizer (PQ) encoding curve shall use the following formula (adapted from SMPTE ST 2084, section 5.3) to map input values x to output values y:

$$y = \left(\frac{c_1 + c_2 x^{m_1}}{1 + c_3 x^{m_1}}\right)^{m_2}$$

where

$$m_1 = {}^{2610}/_{4096} \times \frac{1}{4} = 0.1593017578125$$

$$m_2 = {}^{2523}/_{4096} \times 128 = 78.84375$$

$$c_1 = {}^{3424}/_{4096} = 0.8359375 = c_3 - c_2 + 1$$

$$c_2 = {}^{2413}/_{4096} \times 32 = 18.8515625$$

$$c_3 = {}^{2392}/_{4096} \times 32 = 18.6875$$

The input and output values are normalized per SMPTE ST 2084.

The size of the metadata value shall be 0.

B.9.8 Hybrid Log-Gamma Encoding Curve

The hybrid-log gamma encoding curve shall use the formula specified in Recommendation ITU-R BT.2100-2 Table 5 for the optical-electronic transfer function (OETF) to map input values x to output values y.

The formula is:

If $\mathrm{x} > 1/12$, then $\mathrm{y} = \mathrm{a}\ln(12\,\mathrm{x} - \mathrm{b}) + \mathrm{c}$

otherwise $\mathrm{y} = \sqrt{3\,x}$

where $\mathrm{a} = 0.17883277$, $\mathrm{b} = 1 - 4\,\mathrm{a} = 0.28466892$, and $\mathrm{c} = 0.5 - \mathrm{a}\ln(4\,\mathrm{a}) = 0.55991073$.

The size of the metadata value shall be zero.

Input and output values are as specified in Recommendation ITU-R BT.2100-2.

B.9.9 Encoding Curve Processing (Informative)

The four combinations of inverse encoding curve processing that can be performed by the decoding process on the component arrays decoded from the bitstream are listed in Table B.6.

Table B.6 — Four combinations of encoding curve processing.

Color Space Metadata Item (metadata tag ‘CLSY’) Present in the Bitstream	Encoding Curve Metadata Item Present in the Bitstream	Transfer Function that Applies to the Component Arrays in the Bitstream
No	No	Default
No	Yes	Forward encoding curve
Yes	No	Transfer function as specified by the color space metadata item (metadata tag ‘CLSY’)
Yes	Yes	Forward encoding curve

B.10 Color Difference Component Co-Siting

The metadata item for the color difference component co-siting (metadata tag 'CDCS') specifies the spatial relationship between the luma samples and the color difference component samples in a color difference component image.

The metadata value of the color difference component co-siting metadata tuple shall be one of the values listed in Table B.7 as defined in SMPTE ST 377-1, section G.2.29.

Table B.7 — Metadata values for the color difference component co-siting.

Metadata Value	Co-siting	Description
0	coSiting	The first luma sample of the image is co-sited with the first color difference sample(s), as in ITU-R Rec 601, SMPTE ST 314 4:1:1 or MPEG-2 4:2:2.
1	horizontal mid-point	The color sample is sited at the point horizontally midway between the luma sample on each line.
2	threeTap	Reserved
3	Quincunx	Color samples are sited at the point midway between two adjacent luma samples on two adjacent lines, as in MPEG-1 4:2:0
4	Rec601	Color samples are known to be sited in accordance with ITU-R Rec 601, SMPTE ST 274 and SMPTE ST 296
5	line alternating	The first luma sample is co-sited with the first C'_2 color difference component sample. C'_1 color difference component samples are collocated with C'_2 samples, but appear on alternating lines as in IEC 61834-2 625-50 signals.
6	vertical midpoint	The color sample is sited at the point vertically midway between the luma sample on each column, as in MPEG-2 4:2:0.

NOTE 1: Metadata values 1 (coSiting) and 4 (Rec 601-7) for co-siting in Table B.7 are equivalent. The metadata value 4 in Table B.7 is deprecated (see SMPTE ST 377-1, section G.2.29).

NOTE 2: The C'_1 and C'_2 color differences in Table B.7, defined in SMPTE ST 2073-3, correspond to the Cb and Cr samples as used in SMPTE ST 377-1.

B.11 Frame Rate

The frame rate is expressed as the ratio of the value of the frame rate metadata tuple (metadata tag 'RATE') divided by the value of the time scale metadata tuple (metadata tag 'SCAL'). The values of the frame rate and time scale presented by the frame rate and time scale metadata tuples shall be divided by the least common denominator.

EXAMPLE: The frame rate of 30/1.001 is specified as 30000/1001.

Each of the metadata tuples shall use one of the integer data types defined in section 7.3.

B.12 Frame Number and Timecode

The frame number corresponding to the image represented in the bitstream shall be represented as a 32-bit unsigned integer in the metadata value of the frame number metadata tuple (metadata tag 'FRMN').

The metadata value of the time base metadata tuple (metadata tag ‘TIMB’) shall be the count modulus used for the video sequence. If the time base metadata item is not present in the metadata chunk, then the count modulus shall be 24.

NOTE 1: The count modulus is not necessarily the same as the frame rate (section B.11). Further information about count modulus can be found in RDD 46.

Applications should use the frame rate metadata item (section B.11) to represent the actual frame rate of a video stream.

The value of timecode metadata items shall be represented as character strings in the format specified by SMPTE ST 258, section 8.

The metadata value of the timecode origin (metadata tag ‘TIMS’) shall be the timecode corresponding to the frame number origin. If the timecode origin metadata item is not present in the metadata chunk, then the timecode origin shall be “00:00:00:00”.

The metadata value of the frame number origin (metadata tag ‘FRMZ’) shall be the number of the first frame in the video sequence. If the frame number origin metadata item is not present in the metadata chunk, then the frame number origin shall be 0.

The metadata value of the drop frame metadata tuple (metadata tag ‘TIMD’) shall be 0 or 1. If the value of the drop frame metadata tuple is 0, then the time code shall be non-drop frame timecode; otherwise, the timecode shall be drop-frame. If the drop frame metadata tuple is not present in the metadata chunk, then time shall be non-drop frame.

The timecode corresponding to the frame number (metadata tag ‘FRMN’) can be computed by subtracting the frame number origin (metadata tag ‘FRMZ’) from the frame number and adding this value to the timecode origin (metadata tag ‘TIMS’) taking into account the counting rules of the count modulus (metadata tag ‘TIMB’) and the drop frame flag (metadata tag ‘TIMD’).

NOTE 2: The interpretation of the timecode origin is application dependent and does not necessarily refer to an absolute time scale.

NOTE 3: The issue of frame rate versus field rate is not applicable to VC-5 since fields are represented using two layers, one for each field, within a single frame. For further information about the representation of fields in VC-5 refer to SMPTE ST 2073-5.

Variable frame rates are not supported by this standard.

B.13 Layer Descriptive Metadata

Each layer in a VC-5 bitstream that conforms to SMPTE ST 2073-5 can be described by a layer descriptive metadata tuple with metadata tag ‘LAYR’.

The metadata value of a layer descriptive metadata tuple shall comprise two metadata tuples that specify the number of the layer and metadata that describes the layer. The nested metadata tuples that represent the layer number and description are called parameter tuples to distinguish them from other tuples that represent metadata items and objects.

The metadata tag of the layer number parameter tuple shall be ‘LAYN’. The layer number parameter shall use an integer data type as defined in section 7.3.

The metadata tag of the layer description parameter tuple shall be ‘LAYD’.

NOTE: The data type of the layer description parameter tuple is not specified in this standard but could, for example, be a character string, XML, four character code (FourCC), UUID, SMPTE Universal Label (UL) or any data type listed in Table 4 as appropriate to the application.

B.14 Color Profile

The International Color Consortium (ICC) color profile is defined in ISO/IEC 15076-1:2010.

An ICC color profile can be described by a color profile metadata tuple with metadata tag ‘ICCP’.

The metadata value of a color profile tuple shall be a single ICC color profile.

If both the color space (metadata tag ‘CLSY’) and ICC profile (metadata tag ‘ICCP’) are present in the bitstream for an image or image section, then the color space and ICC profile shall represent the same color space.

Annex C Streaming Data (Normative)

C.1 Streaming Data Overview

This annex specifies a metadata format for representing streaming data samples such as time series data, in a VC-5 bitstream.

Streaming data samples and metadata shall be represented in the bitstream as metadata tuples as described in section 7. The streaming data shall be described using device information (section C.3) and stream information (section C.4).

C.2 Streaming Data Hierarchy

Streaming data tuples are organized into a hierarchy of nested tuples

Sensor device (metadata tag ‘DEVC’)

Data streams within each sensor device (metadata tag ‘STRM’)

Data samples within each data stream.

A detailed example is presented in section C.6.

C.3 Sensor Device Information

The metadata tags for sensor device information shall be as listed in Table C.1.

The device structure metadata item (metadata tag ‘DEVC’) shall indicate the start of sample data streams associated with that device up to the next device structure metadata item.

If present, the device ID (metadata tag ‘DVID’) and device name (metadata tag ‘DVNM’) metadata items shall be nested within the device structure metadata tuple.

Table C.1 — Metadata items for sensor device information.

Metadata Tag	Data Type	Description
‘DEVC’	0	Device structure metadata tuple signaling the start of information for a device
‘DVID’	‘F’, ‘G’, or ‘U’	Device ID that uniquely identifies the device
‘DVNM’	‘c’, ‘u’, or ‘w’	Device name provided as a character string

C.4 Data Stream Information

The metadata tags for sample stream information shall be as listed in Table C.2.

The data stream metadata item (metadata tag ‘STRM’) shall indicate the start of time series data samples associated with that data stream up to the next data stream or device structure metadata item.

If present, the data stream ID (metadata tag ‘STID’), data stream name (metadata tag ‘STNM’), scale factor (metadata tag ‘SCAL’), data stream sample units (metadata tag ‘SIUN’ or ‘UNIT’), data stream time offset (metadata tag ‘TIMO’), and structured data type (metadata tag ‘TYPE’) metadata items shall be nested within the data stream metadata tuple.

Table C.2 — Metadata items for data stream information.

Metadata Tag	Data Type	Description
‘STRM’	0	Data stream tuple signaling the start of information for a data stream
‘STID’	‘F’, ‘G’, ‘U’ or integer data type	Stream ID that uniquely identifies the stream within the device
‘STNM’	‘c’, ‘u’, or ‘w’	Stream name provided as a character string
‘SCAL’	Numerical	Scale factor applied to the data samples during encoding
‘SIUN’	‘c’, ‘u’, or ‘w’	Data stream sample units (SI)
‘UNIT’	‘c’, ‘u’, or ‘w’	Data stream sample units (application-specific)
‘TIMO’	Numerical	Data stream time offset
‘TYPE’	‘c’	Structured data type as described below

The data samples represented in the bitstream shall be divided by the scale factor (metadata tag ‘SCAL’) to restore the values to their original sample units.

Within each data stream metadata tuple, the units of the data samples shall be represented using no more than one of the following metadata items nested in the data stream metadata tuple (metadata tag ‘STRM’):

1. Data stream sample units (metadata tag ‘SIUN’) that use the SI system of measurement as defined in NIST Special Publication 330 and 811, or

2. Data stream sample units (metadata tag ‘UNIT’) that are application-specific and out of scope for this standard.

Each data stream metadata item should include either a data stream sample units (SI) metadata item or a data stream sample units (application-specific) metadata item. The intent is to encourage the use of a standard system of measurement when appropriate while allowing the use of non-standard units.

EXAMPLE 1: Engine RPM can be stored as an SI Unit (rad/s) in a data stream sample units metadata item, or it can be stored as RPM in a data stream sample units (application-specific) metadata item.

The structured data type as listed in Table C.2 is used to specify the representation of streaming data samples that comprise multiple values with different data types. The metadata tuple value of a structured data type (metadata tag ‘TYPE’) shall be a character string of data type codes as specified in Table 4 in the order in which the samples occur in a structured data sample. Multiple contiguous instances of the same data type code may be represented by the data type code followed by the number of instances of that data type as an integer represented in decimal numerals.

EXAMPLE 2: A structured data type comprising two half-floats followed by two signed short integers can be specified by any one of the strings ‘hhss’, ‘h2s2’, ‘h2ss’, or ‘hhs2’.

A vector-valued data sample should be represented by specifying the data type of all elements in the vector as the data type of the metadata tuple that represents the sample vector. The number of elements in the sample vector should be represented by specifying the size of the metadata tuple as the product of the number of elements in each vector times the size of each vector element (in bytes).

C.5 Streaming Data Samples

Each metadata tuple for streaming data samples shall contain zero or more data samples from the data stream signaled by the data stream tuple (metadata tag 'STRM').

The data type of structured data samples shall be specified using a metadata tuple for a structured data type (metadata tag 'TYPE'), otherwise the data type shall be as specified by one of the data types listed in Table 4.

Each metadata tuple for streaming data samples shall be nested within the data stream metadata tuple (metadata tag 'STRM) for that data stream. Metadata tuples for streaming data samples shall follow the data stream metadata tuples listed in Table C.2.

The metadata tag for each streaming data sample is application-specific. Since data samples are nested within a hierarchy of device and stream metadata tuples, metadata tag conflicts are unlikely.

NOTE: The data type of streaming data samples is not restricted to a numerical data type.

C.6 Streaming Data Example (Informative)

Example of the nested structure of streaming data samples within data stream information within device information using the notation described in Annex A.

The following XML represents a telemetry device with an accelerometer data stream and a gyroscope data stream.

The accelerometer data stream is a sequence of short signed integers for the linear acceleration in each of the three Cartesian directions. The size of each data sample is 6 bytes. Since the data type is short signed integer (2 bytes), the size indicates that each data sample is a 3-vector of short signed integers. The values for the acceleration in each direction have been multiplied by a scale factor (metadata tag 'SCAL') so that the acceleration can be represented using integers instead of floating-point values.

The gyroscope data stream is a sequence of floating-point numbers for the angular velocity around each of the three Cartesian axes. The size of each data sample is 12 bytes. Since the data type is single-precision floating-point (4 bytes), the size indicates that each data sample is a 3-vector of single-precision floating-point numbers.

Typically, the repeat count would be many data samples (for example, 100 vectors per data stream tuple), but in this example there are only two vectors so the repeat count is 2.

```
<vc5:tuple tag="DEVC" type="0">
    <vc5:tuple tag="DVID" type="F" value="IMUA" />
    <vc5:tuple tag="DVNM" type="c">Inertial Measurement Unit Type A</vc5:tuple>
    <vc5:tuple tag="STRM" type="0">
        <vc5:tuple tag="STID" type="L" value="1" />
        <vc5:tuple tag="STNM" type="c">Accelerometer Z X Y</vc5:tuple>
        <vc5:tuple tag="SCAL" type="s" value="480" />
        <vc5:tuple tag="SIUN" type="c" value="m/s2" />
        <vc5:tuple tag="ACCL" type="s" size="6" count="2">
            696 1104 643 672 1200 840
        </vc5:tuple>
    </vc5:tuple>
    <vc5:tuple tag="STRM" type="0">
        <vc5:tuple tag="STID" type="L" value="2" />
        <vc5:tuple tag="STNM" type="c">Gyroscope X Y Z</vc5:tuple>
        <vc5:tuple tag="UNIT" type="c" value="deg/s" />
        <vc5:tuple tag="GYRO" type="f" size="12" count="2">
            1.0 2.0 1.35 1.05 2.1 1.45
        </vc5:tuple>
    </vc5:tuple>
</vc5:tuple>
```


Annex D Dark Metadata (Normative)

D.1 Dark Metadata Tags and Data Types

The metadata tags and data types of the metadata items used to represent dark metadata shall be as listed in Table D.1.

Table D.1 — Metadata tags and data types for dark metadata.

Metadata Tag	Data Type	Description
‘VENI’	‘F’, ‘G’, or ‘U’	Identification code for dark metadata (section D.2)
‘VENS’	‘c’, ‘u’, or ‘w’	Identification string for dark metadata (section D.3)
‘VEND’	‘B’	Block of dark metadata (section D.4)

D.2 Dark Metadata Identification Code

The data type of the dark metadata identification code shall be one of the schemes for representing keys listed in Table D.1.

D.3 Dark Metadata Identification String

The data type of the dark metadata identification string shall be one of the character string data types listed in Table D.1.

The vendor identification string should be a short printable character string suitable for identifying the type pf dark metadata to the user of an application. The grammar and vocabulary of the vendor identification string is not specified by this standard.

NOTE: If the vendor identification code is a SMPTE Universal Label for a registry entry, then the vendor identification string could be the description from the registry entry.

D.4 Dark Metadata Item

The payload of dark metadata shall be represented in the bitstream by a metadata tuple with the metadata tag listed in Table D.1. The byte order of the metadata value is out of scope.

NOTE: The term dark metadata does not mean that the metadata is opaque or encrypted, but only means that the syntax and semantics of the metadata is not documented in a published standard.

The byte order of the value in the dark metadata tuple is out of scope.

Annex E XMP Metadata (Normative)

Extensible metadata platform (XMP) is defined in ISO 16684-1:2012.

The contents of an XMP file can be embedded in a VC-5 bitstream as an XMP metadata object with metadata tag and data type as specified in Table 5.

The XMP metadata object comprises one or more of the metadata items listed in Table E.1.

Table E.1 — Metadata items aggregated by an XMP metadata object.

Metadata Tag	Data Type	Description	Required?
‘XMPd’	‘x’	XMP file contents	Yes
‘PATH’	‘c’, ‘u’, or ‘w’	Pathname of the original XMP file	No
‘FCDT’	‘c’, ‘u’, or ‘w’	Creation date of the original XMP file	No
‘FMDT’	‘c’, ‘u’, or ‘w’	Modification date of the original XMP file	No

A separate XMP metadata object shall be used to represent the XMP metadata from each XMP file. The contents of the XMP file shall be copied exactly into the metadata value of the XMP file contents metadata item (metadata tag ‘XMPd’). If the bitstream uses image sections, then the XMP metadata objects associated with an image shall be contained within the corresponding image section.

The pathname (metadata tag ‘PATH’) should be formatted as a URI defined in IETF RFC 3986.

The creation and modification dates (metadata tags ‘FCDT’ and ‘FMDT’) should conform to ISO 8601.

The mapping from the pathname of the original XMP file to a valid pathname on the computer system running a decoder implementation is out of scope.

NOTE 1: The intent is that the XMP file can be recreated by the image repacking process.

NOTE 2: The intent of providing metadata items for the file name and attributes is to allow the XMP metadata to be output by the image repacking process to a file with the same name and attributes.

Annex F DPX Metadata (Normative)

The file header from a DPX file as specified in SMPTE ST 268-1 and SMPTE ST 268-2 can be embedded in a VC-5 bitstream as a metadata object with metadata tag and data type as specified in Table 5.

The image repacking process should output the image to the same location in the resulting DPX file as specified by field 2x.12 in the DPX file header.

The DPX metadata object comprises one or more of the metadata items listed in Table F.1.

Table F.1 — Metadata items aggregated by a DPX metadata object.

Metadata Tag	Data Type	Description	Required?
‘DPXh’	‘B’	DPX file header	Yes
‘PATH’	‘c’, ‘u’, or ‘w’	Pathname of the original DPX file	No
‘FCDT’	‘c’, ‘u’, or ‘w’	Creation date of the original DPX file	No
‘FMDT’	‘c’, ‘u’, or ‘w’	Modification date of the original DPX file	No

A separate DPX metadata object shall be used to represent the DPX file header from each DPX file. If the bitstream uses image sections, then the DPX metadata object associated with an image shall be contained within the corresponding image section.

There should be at most one DPX metadata object associated with each image represented in the bitstream.

The DPX file header shall be copied exactly into the metadata value of the DPX file header metadata item (metadata tag ‘DPXh’), starting with the first byte in the file. A DPX file can allocate a larger file header than is actually used. The size of the metadata value of the DPX file header metadata item shall be at least as large as the portion of the allocated file header that contains valid data.

The pathname (metadata tag ‘PATH’) should be formatted as a URI defined in IETF RFC 3986.

The creation and modification dates (metadata tags ‘FCDT’ and ‘FMDT’) should conform to ISO 8601.

The mapping from the pathname of the original DPX file to a valid pathname on the computer system running a decoder implementation is out of scope.

NOTE 1: The intent is that the file header of a DPX file that contains the image encoded into the VC-5 bitstream can be preserved so that the decoded component arrays can be output to a DPX file by the image repacking process with the identical DPX file header as the source image, except that the header of the output file can have a larger allocation.

NOTE 2: The intent of providing metadata items for the file name and attributes is to allow the DPX metadata to be output by the image repacking process to a file with the same name and attributes.

Annex G MXF Annex F and G Essence Descriptors (Normative)

The picture essence descriptors from an MXF file as specified in SMPTE ST 377-1:2011 section F-4 can be embedded in a VC-5 bitstream as a metadata object with metadata tag and data type as specified in Table 5.

The MXF picture essence descriptor metadata object comprises one or more of the metadata items listed in Table G.1.

Table G.1 — Metadata items aggregated by an MXF metadata object.

Metadata Tag	Data Type	Description	Required?
'MXFd'	'B'	MXF picture essence descriptor	Yes

The picture essence descriptors embedded in a VC-5 bitstream shall be zero or more of the picture essence descriptors listed in Table G.2.

Table G.2 — Picture essence descriptors that may be embedded in a VC-5 bitstream.

Picture Essence Descriptor	Section Reference in SMPTE ST 377-1:2011
Generic Picture Essence Descriptor	F.4.1
CDCI (Color Difference Component Image) Picture Essence Descriptor	F.4.2
RGBA (Red Green Blue Alpha) Picture Essence Descriptor	F.4.3

Annex H ACES Attributes (Normative)

The header from an ACES image container file specified in SMPTE ST 2065-4 can be embedded in a VC-5 bitstream as a metadata object with metadata tag and data type as specified in Table 5.

The ACES metadata object comprises one or more of the metadata items listed in Table H.1.

Table H.1 — Metadata items aggregated by an ACES metadata object.

Metadata Tag	Data Type	Description	Required?
‘ACEh’	‘B’	ACES image container file header	Yes
‘PATH’	‘c’, ‘u’, or ‘w’	Pathname of the original ACES file	No
‘FCDT’	‘c’, ‘u’, or ‘w’	Creation date of the original ACES file	No
‘FMDT’	‘c’, ‘u’, or ‘w’	Modification date of the original ACES file	No

A separate ACES metadata object shall be used to represent the header from each ACES image container file. If the bitstream uses image sections, then the ACES metadata object associated with an image shall be contained within the corresponding image section.

There should be at most one ACES metadata object associated with each image represented in the bitstream.

The metadata value of an ACES attributes metadata item (metadata tag ‘ACEh’) shall comprise a contiguous sequence of one or more ACES attribute structures, followed immediately by the terminating null byte (0x00), as defined in SMPTE ST 2065-4 Section 7.5.

The pathname (metadata tag ‘PATH’) should be formatted as a URI defined in IETF RFC 3986.

The creation and modification dates (metadata tags ‘FCDT’ and ‘FMDT’) should conform to ISO 8601.

The mapping from the pathname of the original ACES file to a valid pathname on the computer system running a decoder implementation is out of scope.

NOTE 1: The intent is that the file header of an ACES image container file that contains the image encoded into the VC-5 bitstream can be preserved so that the decoded component arrays can be output to an ACES image container by the image repacking process with the identical ACES header fields and attributes as the source image.

NOTE 2: The intent of providing metadata items for the file name and attributes is to allow the ACES metadata to be output by the image repacking process to a file with the same name and attributes.

Annex I ALE Metadata (Normative)

ALE is a file format for the exchange of metadata in post-processing.

ALE stores metadata in a text file. Each row corresponds to a metadata record. Typically, each metadata record corresponds to a media file. In each row, columns are delineated by tab characters or commas. Each column corresponds to a type of metadata. Column headings identify the type of metadata.

The contents of an ALE file can be embedded in a VC-5 bitstream as an ALE metadata object with metadata tag and data type as specified in Table 5.

The ALE metadata object comprises one or more of the metadata items listed in Table I.1.

Table I.1 — Metadata items aggregated by an ALE metadata object.

Metadata Tag	Data Type	Description	Required?
‘ALEd’	‘c’, ‘u’, or ‘w’	ALE file contents	Yes
‘PATH’	‘c’, ‘u’, or ‘w’	Pathname of the original ALE file	No
‘FCDT’	‘c’, ‘u’, or ‘w’	Creation date of the original ALE file	No
‘FMDT’	‘c’, ‘u’, or ‘w’	Modification date of the original ALE file	No

A separate ALE metadata object shall be used to represent the ALE metadata from each ALE file. The contents of the ALE file shall be copied exactly into the metadata value of the ALE file contents metadata item (metadata tag ‘ALEd’). If the bitstream uses image sections, then the ALE metadata objects associated with an image shall be contained within the corresponding image section.

The pathname (metadata tag ‘PATH’) should be formatted as a URI defined in IETF RFC 3986.

The creation and modification dates (metadata tags ‘FCDT’ and ‘FMDT’) should conform to ISO 8601.

The mapping from the pathname of the original ALE file to a valid pathname on the computer system running a decoder implementation is out of scope.

NOTE 1: The intent is that the ALE file can be recreated by the image repacking process.

NOTE 2: The intent of providing metadata items for the file name and attributes is to allow the ALE metadata to be output by the image repacking process to a file with the same name and attributes.

Annex J Dynamic Metadata for Color Volume Transform (Normative)

The dynamic metadata for color volume transform (DMCVT) as specified in SMPTE ST 2094-2 can be embedded in a VC-5 bitstream as a metadata object with metadata tag and data type as specified in Table 5.

The value of the DMCVT metadata object comprises the metadata items listed in Table J.1 and nothing more.

Table J.1 — Metadata items aggregated by a DMCVT metadata object.

Metadata Tag	Data Type	Description	Required?
‘CVTS’	‘U’	SMPTE Universal Label (UL) that identifies the DMCVT generic or application set as defined in SMPTE ST 2094-2	Yes
‘CVTD’	‘B’	Contiguous sequence of SMPTE KLV tuples for the dynamic color volume transform metadata and nothing more	Yes

The metadata item for the universal label that identifies the generic or application set (metadata tag ‘CVTS’) should precede the metadata item for the color volume transform metadata (metadata tag ‘CVTD’).

NOTE: The value of the universal label metadata item (metadata tag ‘CVTS’) is a leaf node.

The KLV tuples that comprise the value of the color volume transform metadata (metadata tag ‘CVTD’) shall be as defined in SMPTE ST 2094-2 in the table corresponding to the DMCVT generic or application set (metadata tag ‘CVTS’).

Bibliography (Informative)

SMPTE RDD 31:2014 Deferred Demosaicing of an ARRIRAW Image File to a Wide Gamut Logarithmic Encoding.

SMPTE RDD 46:2019 SMPTE Timecode Extensions – Relationships to Higher Rates and Date-Time.

XMP Specification Part 1: Data Model, Serialization, and Core Properties, April, 2012.

ALEXA Log C Curve: Usage in VFX (PDF).

Charles Poynton, Digital Video and HD: Algorithms and Interfaces (2nd Edition), Morgan Kaufmann, 2012, Chapter 27.

IETF RFC 4648, The Base16, Base32, and Base64 Data Encodings, October 2006.

ARIB STD-B67 Essential parameter values for the extended image dynamic range television (EIDRTV) system for programme production, Version 1.0, July 3. 2015.

