SMPTE STANDARD

VC-5 Video Essence — Part 3: Image Formats

Table of Contents

Approved
January 2, 2015

Figures

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.

SMPTE ST 2073-3 was prepared by Technology Committee 10E.

Intellectual Property

SMPTE draws attention to the fact that it is claimed that compliance with this Standard may involve the use of one or more patents or other intellectual property rights (collectively, "IPR"). The Society takes no position concerning the evidence, validity, or scope of this IPR.

Each holder of claimed IPR has assured the Society that it is willing to License all IPR it owns, and any third party IPR it has the right to sublicense, that is essential to the implementation of this Standard to those (Members and non-Members alike) desiring to implement this Standard under reasonable terms and conditions, demonstrably free of discrimination. Each holder of claimed IPR has filed a statement to such effect with SMPTE. Information may be obtained from the Director, Standards & Engineering at SMPTE Headquarters.

Attention is also drawn to the possibility that elements of this Standard may be subject to IPR other than those identified above. The Society shall not be responsible for identifying any or all such IPR.

Introduction

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

The VC-5 Elementary Bitstream defined in SMPTE ST 2073-1 encodes arrays of components as independent channels and does not specify the type of components that are represented in the bitstream, but defines an extensible syntax for the elements that comprise a bitstream. This standard extends the bitstream syntax with new elements that describe the format of the image represented by a bitstream.

The image formats covered by this standard are images that can be modeled as a single rectilinear grid of samples, called a sample array, with a fixed number of components per sample, including image formats such as R′G′B′(A) and Y′C′$_1$C′$_2$(A) without sub-sampled color differences.

For the purposes of this standard, the C′$_1$ and C′$_2$ color difference components correspond to the blue color difference and red color difference, respectively.

A sample array can model any Color Filter Array (CFA) sensor, such as Bayer, that consists of a repetition of rectilinear pattern elements with the same structure.

1 Scope

This standard specifies extensions to the VC-5 Elementary Bitstream defined in SMPTE ST 2073-1 for representing R′G′B′(A) images, $Y'C'_1C'_2$(A) images without color difference component sub-sampling, and Color Filter Array (CFA) images including Bayer image formats. This standard specifies the default color space for R′G′B′ and $Y'C'_1C'_2$ color values.

This standard extends SMPTE ST 2073-1 by adding new parameters and chunk elements:

The **ImageFormat** parameter specifies the type of image: R′G′B′(A), $Y'C'_1C'_2$(A), CFA, or Bayer;

The **PatternWidth**, **PatternHeight**, and **ComponentsPerSample** parameters specify the structure of CFA images, including Bayer images;

The **MaxBitsPerComponent** parameter specifies an upper bound on the number of bits needed to represent all values in the component arrays present in the bitstream;

The **VendorSpecificData** chunk element allows private data to be embedded in a VC-5 bitstream;

The **InverseTransform**, **InverseTransform16**, and **InversePermutation** chunk elements specify how to transform and reorder decoded component arrays into the same color space and order as originally presented to the encoder;

The **UniqueImageIdentifier** chunk element specifies the image sequence and position within the sequence of an image represented by a VC-5 bitstream.

This standard extends SMPTE ST 2073-1 by defining three stages for transforming decoded component arrays into images: inverse component transform, inverse component permutation, and image repacking process.

This standard extends SMPTE ST 2073-1 to specify levels of performance for decoder implementations.

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; followed by formal languages; then figures; and then any other language forms.

3 Normative References

The following standards contain provisions that, 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.

SMPTE ST 330:2011, Unique Material Identifier (UMID)

SMPTE ST 2073-1:2014, VC-5 Video Essence — Part 1: Elementary Bitstream

IEEE 802-2001, Standard for Local and Metropolitan Area Networks: Overview and Architecture.

4 Terms and Definitions

For the purposes of this standard, the following definitions shall apply:

4.1 color difference component

Color component that represents the difference between two colors.

4.2 color difference component sub-sampling

Color difference components with less spatial resolution than the Y′ components.

4.3 component vector

One-dimensional vector of scalar values.

4.4 Y′ component

Weighted sum of R′G′B′ color components that are optionally gamma-corrected.

4.5 pattern element

Two-dimensional array of samples used to tessellate the sample array.

Note: The pattern element normally contains just a few samples.

4.6 Sample

Element in an array of samples, each element comprising one or more component values.

4.7 sample array

Two-dimensional array of samples.

4.8 sample array dimensions

Width and height of the sample array in units of samples.

4.9 sample array height

Number of lines (rows) in a sample array.

4.10 sample array width

Number of samples per line (row) in a sample array.

4.11 signed integer

Two's complement integer that may be positive, negative, or zero.

4.12 unsigned integer

Integer that may be positive or zero.

4.13 VC-5 bitstream

Bitstream that is compliant with one or more parts of SMPTE ST 2073.

5 Notation

This document uses the notation defined in SMPTE ST 2073-1 VC-5 Elementary Bitstream.

The value of the operation $\mathrm{clamp}(x, l, u)$ is l if x is less than l, u if x is greater than u, and otherwise equal to x.

6 Overview (Informative)

This standard extends the elementary bitstream with syntax elements that describe how common image formats are represented in VC-5 bitstreams.

Each of the image formats covered in this document is modeled as a single sample array with one or more components per sample. Sample arrays are described in Section 7.

Any image that can be modeled as a single sample array can be unpacked into component arrays with one component from each sample in each array (Section 8). This standard defines the inverse component transform for reversing the effect of color space transformation applied by the encoder to improve compression (Section 8.5) and defines the inverse component permutation for reordering the component values into the same order as originally presented to the encoder (Section 8.6).

Section 9 describes the image formats defined in this standard: R′G′B′ images with an optional alpha channel (Section 9.1), $Y'C'_1C'_2$ images with an optional alpha channel, but no sub-sampling (Section 9.2), Color Filter Array (CFA) images (Section 9.3), and Bayer images (Section 9.4).

Extensions to the bitstream syntax that are defined by this standard are described in Sections 10 and 11.

The default colorimetry for images with R′G′B′ or $Y'C'_1C'_2$ color components is defined in Annex C.

Two new stages in the decoding process for applying the inverse component transform and reordering the components using the inverse component permutation are described in Annex D. The inverse transform and permutation followed by the image repacking process (also described in Annex D) allows the component arrays output by the decoding process as defined in SMPTE ST 2073-1 (VC-5 Elementary Bitstream) to be repacked into an image.

7 Image Structure

7.1 Sample Arrays (Informative)

Images that do not have color difference component sub-sampling can be modeled as a single sample array. A sample consists of one or more components at the same location in the sample array. All samples contain the same number and type of components in the same order. Components may be color components or other data such as alpha values for compositing or disparity values. Sometimes, in other documents, the use of the term pixel corresponds to the use of the term sample in this document.

The dimensions of a sample array are in units of samples.

Sample arrays have W samples per row and H rows of samples for a total of M samples:

$$M = W \times H$$

Each sample has C components per sample for a total of N components per sample array:

$$N = M \times C$$

Multiple image formats can map to the same set of component arrays input to the encoder. The image repacking process cannot output the original image format input to the image unpacking process without additional information provided to the decoder by the application.

7.2 Pattern Elements (Informative)

The sample array can be partitioned into a repeating pattern of samples as shown in Figure 1. The repeating pattern of samples is called a pattern element. The sample order is the same in every pattern element.

Each pattern element has w samples per row and h samples per column for a total of m samples per pattern element:

$$m = w \times h$$

Each sample has C components per sample for a total of n components per pattern element:

$$n = m \times C$$

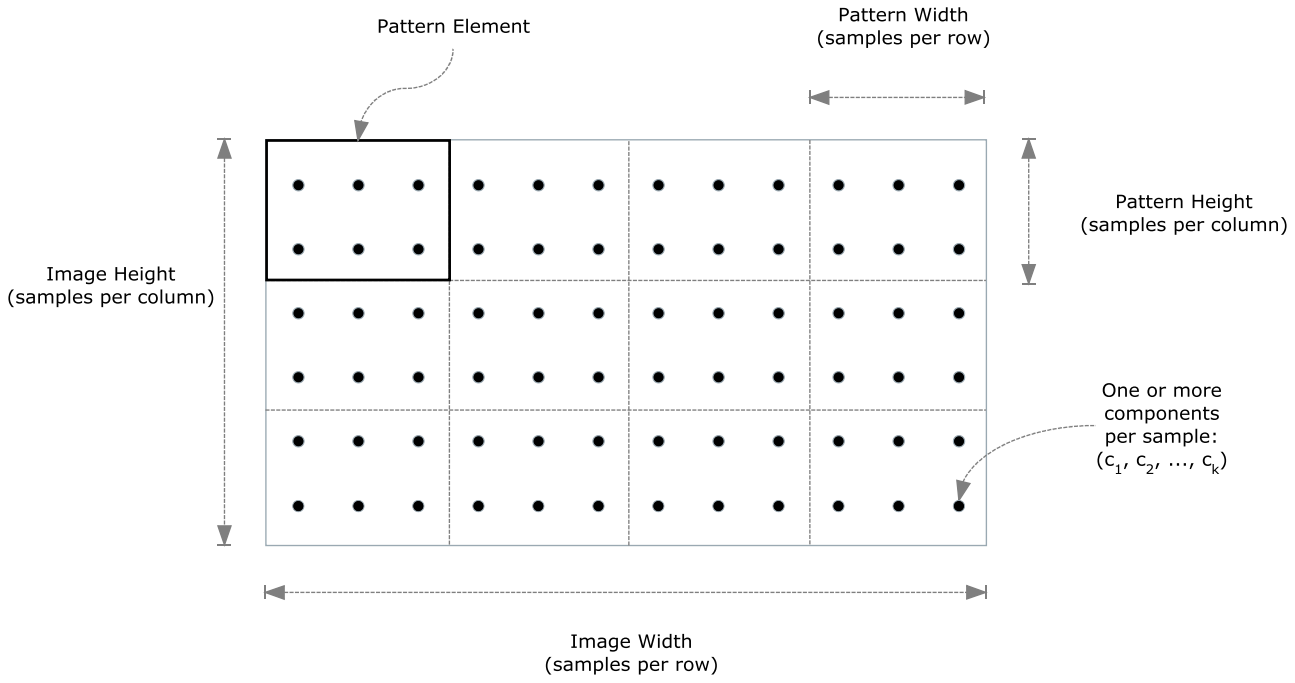

Figure 1 – A sample array consists of a repeating pattern of samples (pattern elements)

7.3 Pattern Element Dimensions

A sample array shall be partitioned into a repeating pattern of rectangular regions called pattern elements. (See the example in Figure 1.)

Each pattern element in the sample array shall be the same width and height and shall consist of the same component types in the same order.

The number of samples in each row of a pattern element shall be **PatternWidth**. The number of sample rows in each pattern element shall be **PatternHeight**. The number of components per sample shall be **ComponentsPerSample**.

The order of the components in each sample shall be the same.

Note: A pattern element is a three-dimensional array of component values (sample row, column within each row, and index to each component value in a sample).

7.4 Image Dimensions

The **ImageWidth** shall equal the number of samples in each row and **ImageHeight** shall equal the number of samples in each column of the sample array regardless of the dimensions of the pattern elements (Figure 1).

The value of **ImageWidth** shall be an integer multiple of **PatternWidth** and the value of **ImageHeight** shall be an integer multiple of **PatternHeight**.

8 Bitstream Structure

8.1 Component Array Count

The value of **ChannelCount** shall equal the product of **PatternWidth** times **PatternHeight** times **ComponentsPerSample**.

Note: **ChannelCount** is equal to n, where n is the number of components per pattern element (Section 7.2).

8.2 Component Array Dimensions

For all component arrays:

(1) The width of the component array shall be **ImageWidth** divided by **PatternWidth**.

(2) The height of the component array shall be **ImageHeight** divided by **PatternHeight**.

Tag-value pairs for **ChannelWidth** and **ChannelHeight** shall not be present in the bitstream.

8.3 Component Vector

A component vector shall comprise the component values in all component arrays at the same row index and column index. The component values in the component vector that is input to the inverse component transform shall be ordered by channel number.

8.4 Inverse Component Transform and Inverse Component Permutation

The inverse component transform shall be applied to every component vector in the decoded component arrays. The inverse component permutation shall be applied to every component vector output by the inverse component transform. See Figure D.1.

8.5 Inverse Component Transform

The inverse component transform shall be applied to the component vector corresponding to every pair of row index and column index in the decoded component arrays.

The output of the inverse component transform shall be a vector of unsigned integer values. The number of values in the output vector shall be the same as the number of values in the component vector input to the inverse component transform.

The inverse component transform is an affine transform implemented with signed integer arithmetic and scaling. Each output component of the inverse component transform shall be computed as:

$$r_i = \left(\sum_{j=0}^{n-1} a_{i,j} c_j'' + b_i \right)$$

$$c_i' = \mathrm{clamp}(\mathrm{ash}(r_i, s_i), 0, 2^{p_i} - 1)$$

where

c_j'' is an input value from the component array input to the inverse component transform,

c_i' is the output value from the component array produced by the inverse component transform,

i is the index of the component value in the output component vector,

j is the index of the component value in the input component vector,

$a_{i,j}$ are the coefficients of an affine transform matrix,

b_i are the coefficients of an affine transform offset (translation vector),

r_i is the intermediate result from the affine transform before scaling by an arithmetic right shift,

s_i is the number of bit positions of arithmetic right shift required to scale the result of the affine transform to the precision of the component array values,

n equals the number of components per pattern element, and

p_i equals the value of **BitsPerComponent** for the output component.

For all image formats except the Bayer image format (Section 9.4), the default inverse component transform shall be the identity component transform. The identity component transform shall be defined as $a_{i,j} = 0$, $b_i = 0$, $s_i = 0$, for all i and j such that $0 \leq i < n$ and $0 \leq j < n$ except $a_{i,j} = 1$ when $i = j$.

If present in the bitstream, the inverse component transform shall be represented by an inverse component transform chunk (Section 11.1.3).

Note: The $\mathrm{ash}()$ function is defined in SMPTE ST 2073-1, Section 5.3.

8.6 Inverse Component Permutation

The inverse component permutation shall be applied to the output vector from the inverse component transform. Refer to Figure D.1.

The inverse component permutation is a one-to-one mapping that for each index i with $0 \leq i < n$, where n is the number of components per pattern element, specifies the new index i' for each component in the vector, with $0 \leq i' < n$, such that each value of i' occurs in the permutation exactly once.

The default inverse component permutation shall be the identity permutation. The identity permutation shall preserve the order of the values in the component vector.

8.7 Mapping Between Pattern Element and Component Vector

The component values in a component vector shall map one-to-one to the component values in a pattern element as follows:

$$i' = i\ w\ C + j\ C + k$$

where:

i is the index of the row of samples in the pattern element with $0 \leq i < h$,

j is the index of the column of samples in the pattern element with $0 \leq j < w$,

k is the index of a component value in a sample with $0 \leq k < C$,

i' is the index of the component value in the component vector with $0 \leq i' < n$,

h is the value of **PatternHeight**,

w is the value of **PatternWidth**, and

C is the value of **ComponentsPerSample**.

9 Image Formats

9.1 R′G′B′(A) Image Format

For the R′G′B′(A) image format, the values of the **PatternWidth**, **PatternHeight**, **ComponentsPerSample**, and **ImageFormat** parameters shall be as specified Table 1.

Table 1 – Parameters for the R′G′B′(A) image format

Parameter Name	Value
PatternWidth	1
PatternHeight	1
ComponentsPerSample	3 or 4
ImageFormat	1

The red (R′), green (G′), and blue (B′) component arrays shall all be present in the bitstream. If and only if **ComponentsPerSample** is 4, then the alpha (A) component array shall be present in the bitstream. Component arrays shall be assigned to channel numbers as listed in Table 2.

Table 2 – Channel number assignment for R′G′B′(A) images.

Component array	Channel Number
G'	0
R′	1
B′	2
A	3

Note: The channel number assignment for the R′G′B′(A) specified in Table 2 implies that the order of the R′G′B′(A) components in the input image is G′R′B′(A) when the composition of the component permutation and the component transform is equivalent to the identity permutation.

Note: An example of an R′G′B′ image with 4:4:4 sub-sampling is shown in Figure 2. Each pattern element is one sample in width and one sample in height. Every dot represents a sample and each sample has three components: R′, G′, and B′.

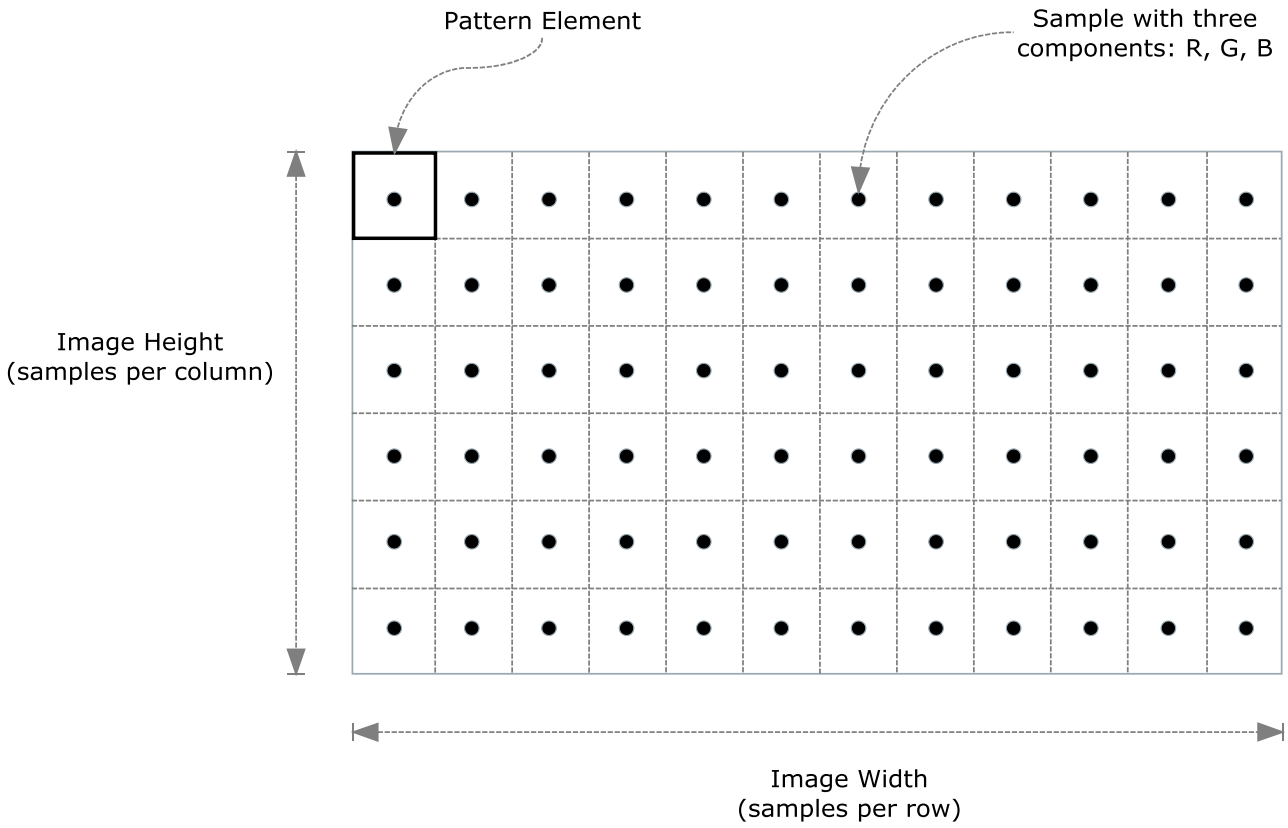

Figure 2 – The sample array for an R′G′B′ image with 4:4:4 sub-sampling

9.2 Y′C′$_1$C′$_2$(A) Image Format

For the Y′C′$_1$C′$_2$(A) image format, the values of the **PatternWidth**, **PatternHeight**, **ComponentsPerSample**, and **ImageFormat** parameters shall be as specified in Table 3.

Table 3 – Parameters for the Y′C′$_1$C′$_2$(A) image format

Parameter Name	Value
PatternWidth	1
PatternHeight	1
ComponentsPerSample	3 or 4
ImageFormat	2

The Y′ component array, C'_1 component array, and C'_2 component array shall all be present in the bitstream. If and only if **ComponentsPerSample** is 4, then the alpha (A) component array shall be present in the bitstream. Component arrays shall be assigned to channel numbers as listed in Table 4.

Table 4 – Channel number assignment for $Y'C'_1C'_2$(A) images

Component array	Channel Number
Y′	0
C'_2	1
C'_1	2
A	3

Note: The channel number assignment for the $Y'C'_1C'_2$(A) image format specified in Table 4 implies that the order of the $Y'C'_1C'_2$(A) components in the input image is $Y'C'_2C'_1$(A) when the composition of the component permutation and the component transform is equivalent to the identity permutation.

9.3 CFA Image Format

A color filter array (CFA) image can be modeled as a single sample array that is partitioned into pattern elements (Section 7.3).

For the CFA image format, the **ImageFormat** tag shall be present in the bitstream header and the value shall be 4. One or both of the **PatternWidth** or **PatternHeight** parameters shall be greater than 1 and the **ComponentsPerSample** shall be 1.

As specified in SMPTE ST 2073-1 (VC-5 Elementary Bitstream), the component arrays from the image unpacking process shall be assigned to consecutive channel numbers in the order in which the components are output by the image unpacking process, starting with channel number zero.

9.4 Bayer Image Format

9.4.1 Bayer Image Parameters

For the Bayer image format, the values of the **PatternWidth**, **PatternHeight**, **ComponentsPerSample**, and **ImageFormat** parameters shall be as specified in Table 5. The value of **BitsPerComponent** shall be the same for all channels.

Table 5 – Parameters for the Bayer image format

Parameter Name	Value
PatternWidth	2
PatternHeight	2
ComponentsPerSample	1
ImageFormat	3

Component arrays shall be assigned to channel numbers as listed in Table 6.

Table 6 – Channel number assignment for Bayer images

Component array	Channel Number
G_{sum}	0
RG	1
BG	2
G_{diff}	3

9.4.2 Inverse Bayer Component Transform

The default inverse component transform shall compute the G_1, G_2, R, and B color components using the following formulas:

$$G_1 = \text{clamp}(G_{\text{sum}} + (G_{\text{diff}} - V), 0, L) \qquad G_2 = \text{clamp}(G_{\text{sum}} - (G_{\text{diff}} - V), 0, L)$$

$$R = \text{clamp}(G_{\text{sum}} + 2 \times (\text{RG} - \text{V}), 0, L) \qquad B = \text{clamp}(G_{\text{sum}} + 2 \times (\text{BG} - \text{V}), 0, L)$$

where

G_1 is the first green component,

G_2 is the second green component,

R is the red component,

B is the blue component,

G_{sum} is the sum of the green components,

G_{diff} is the difference between the green components,

RG is the red difference,

BG is the blue difference,

$V = 2^{(p-1)}$ is the midpoint value,

$L = 2^p - 1$ is the maximum value for the transformed Bayer components, and

p is the value of **BitsPerComponent**.

The order of the Bayer components in the component vector output by the inverse component transform shall be R, G_1, G_2, and B, where R and G_1 are the color components in the first row of the Bayer pattern element in order by column index and G_2 and B are the color components in the second row of the Bayer pattern element in order by column index.

Note: The arrangement of the Bayer components corresponds to the pattern element:

R	G_1
G_2	B

9.4.3 Forward Bayer Component Transform (Informative)

The forward component transform is the inverse of the inverse component transform (Section 9.4.2) and computes the green sum, green difference, red difference, and blue difference from the Bayer color components using the following formulas:

$$G_{\text{sum}} = \text{clamp}(\text{ash}(G_1 + G_2, 1), 0, L) \qquad G_{\text{diff}} = \text{clamp}(\text{ash}(G_1 - G_2 + 2V, 1), 0, L)$$

$$RG = \text{clamp}(\text{ash}(R - G_{\text{sum}} + 2V, 1), 0, L) \qquad BG = \text{clamp}(\text{ash}(B - G_{\text{sum}} + 2V, 1), 0, L)$$

10 Bitstream Syntax

10.1 Bitstream Header

The set of codec state parameters that may be present in the bitstream header defined in SMPTE ST 2073-1 is extended by this standard to include the codec state parameters that are designated as bitstream header parameters in Table A.1.

10.2 Vendor-Specific Data

Zero or more vendor-specific data chunks (Section 11.1.2) shall occur only before the first codeblock.

10.3 Inverse Component Transform

Zero or one inverse component transform chunks (Section 11.1.3) shall occur only before the first codeblock.

10.4 Inverse Component Permutation

Zero or one inverse component permutation chunks (Section 11.1.4) shall occur only before the first codeblock.

10.5 Unique Image Identifier

Zero or one unique image identifier chunks (Section 11.1.5) shall occur only before the first codeblock.

11 Syntax Elements

11.1 Chunk Elements

11.1.1 Chunk Payload Padding

As defined in SMPTE ST 2073-1, a chunk element consists of a chunk header followed by the chunk payload and the chunk payload consists of zero or more segments. SMPTE ST 2073-1 does not distinguish between the portion of the payload that contains data and the remainder of the payload that consists of padding to the nearest segment boundary.

The chunk payload shall consist of chunk payload data followed by chunk payload padding and nothing more.

Chunk payload data shall consist of the contiguous sequence of bits containing data, beginning immediately after the last bit in the chunk header.

Chunk payload padding shall consist of the contiguous sequence of 0-31 bits, beginning immediately after the last bit in the chunk payload data.

11.1.2 Vendor-Specific Data

Vendor-specific data defined by this standard shall be represented by **VendorSpecificData** small chunk elements as specified in Table B.1.

The first three bytes in the chunk payload data of a **VendorSpecificData** small chunk element shall be the vendor ID represented by an Organizationally Unique Identifier (OUI), as defined in IEEE 802. The length and content of the chunk payload data after the vendor ID is not defined by this standard.

Note: The OUI is also known as a MAC Address Block Large (MA-L).

If more than one **VendorSpecificData** small chunk element has the same vendor ID, then the behavior is not defined by this standard.

11.1.3 Inverse Component Transform

11.1.3.1 Inverse Component Transform (Generic)

The inverse component transform shall be represented using either an **InverseTransform** small chunk element or an **InverseTransform16** small chunk element, but not both**.**

The chunk payload data shall consist only of the coefficients, offset, and shift for each output component in order of output component.

The values of $a_{i,j}$, b_i, and s_i of the inverse component transform shall be represented in the chunk payload by a contiguous sequence of bytes in ascending order of index i, and for each index i in ascending order of index j from smallest index to largest index.

11.1.3.2 Inverse Component Transform (8-bit)

The 8-bit representation of the inverse component transform described in Section 8.5 shall be represented by an **InverseTransform** small chunk element as specified in Table B.1.

In the **InverseTransform** small chunk element, each coefficient $a_{i,j}$ and offset b_i shall be represented by an 8-bit signed integer and each arithmetic right shift s_i shall be represented by an 8-bit unsigned integer.

Note: The number of bytes of chunk payload data will be $n^2 + 2n$, where n is the value of **ChannelCount**.

11.1.3.3 Inverse Component Transform (16-bit)

The 16-bit representation of the inverse component transform described in Section 8.5 shall be represented by an **InverseTransform16** small chunk element as specified in Table B.1.

In the **InverseTransform16** small chunk element, each coefficient $a_{i,j}$ and offset b_i shall be represented by a 16-bit big-endian signed integer and each arithmetic right shift s_i shall be represented by a 16-bit big-endian unsigned integer.

Note: The number of bytes of chunk payload data will be $2n^2 + 4n$, where n is the value of **ChannelCount**.

11.1.4 Inverse Component Permutation

The inverse component permutation described in Section 8.6 shall be represented by an **InversePermutation** small chunk element as specified in Table B.1.

The **InversePermutation** chunk payload data shall consist of a sequence of **ChannelCount** unsigned bytes for the component order. The byte at index i' with value i specifies that the input component array value at index i shall be mapped to index i' in the output component array. The value of each byte for the component order shall be in the range [0, **ChannelCount** − 1].

If the inverse component permutation is present in the bitstream, then the value of **ChannelCount** must not be greater than 256; otherwise, the value of **ChannelCount** shall be as defined in SMPTE ST 2073-1.

Note: The representation for the inverse component permutation defined by this standard limits the component vector index to the range [0, 256). If the inverse component permutation is implicitly defined to be the identity permutation by its absence from the bitstream; then the component vector index is not limited by the representation of the inverse component permutation.

11.1.5 Unique Image Identifier

The unique image identifier shall be represented by a **UniqueImageIdentifier** small chunk element as specified in Table B.1.

The chunk payload data of the **UniqueImageIdentifier** small chunk element shall comprise only the following two elements:

(1) UMID that identifies the image sequence that contains the image,

(2) Sequence number that identifies the position of the image in the sequence.

The sequence number shall be a 32-bit big-endian unsigned integer.

If the image is not a member of a sequence of images, then the sequence number should be zero; otherwise, images with the same UMID shall be numbered consecutively and the first image in a sequence shall have sequence number 1.

Note: The sequence number allows a single image to be distinguished from an image that is a member of a video clip or time-lapse sequence that has a length of 1.

Annex A Codec State Parameters (Normative)

A.1 Bitstream Extensions

The codec state parameters defined in SMPTE ST 2073-1 Annex B.1 shall be extended by this standard to include the codec state parameters defined in Annex A.2 of this standard.

The tag-value pairs defined in SMPTE ST 2073-1 Annex B.2 shall be extended by this standard to include the tag-value pairs defined in Annex A.3 of this standard.

A.2 Parameter Descriptions

A.2.1 PatternWidth

The **PatternWidth** parameter shall specify the number of samples per row in each pattern element of the sample array output by the image repacking process. The value shall be in the range [1, min(**ImageWidth**, 255)].

A.2.2 PatternHeight

The **PatternHeight** parameter shall specify the number of rows in each pattern element of the sample array output by the image repacking process. The value shall be in the range [1, min(**ImageHeight**, 255)].

A.2.3 ComponentsPerSample

The **ComponentsPerSample** parameter shall specify the number of components in each sample of the sample array output by the image repacking process. The value shall be in the range [1, $C_{\max}$], where

$C_{\max} = \left\lfloor \frac{256}{h \times w} \right\rfloor$ if the inverse component transform is present in the bitstream, otherwise

$C_{\max} = \left\lfloor \frac{65535}{h \times w} \right\rfloor$

where

h is the value of **PatternHeight** and
w is the value of **PatternWidth.**

A.2.4 Image Format

A tag-value pair for the **ImageFormat** parameter shall be present in the bitstream and the value shall be in the range from 1 to 4, inclusive.

A.2.5 Maximum Bits Per Component

The **MaxBitsPerComponent** parameter shall specify an upper bound on the number of bits per component in all component arrays represented by the bitstream. The value shall be in the same range as defined for the **BitsPerComponent** parameter in SMPTE ST 2073-1.

Note: The **MaxBitsPerComponent** parameter enables a decoder implementation to determine the bitstream level from parameters that are present in the bitstream header without parsing the remainder of the bitstream.

A.3 Tag-Value Pairs

The tag-value pairs that represent the codec state parameters defined by this standard in a VC-5 bitstream shall be as listed in Table A.1, with the tag number and default value as listed in that table. The tag-value pairs defined by this standard shall be header parameters if so specified in Table A.1.

Table A.1 – Codec state parameters defined by this standard

Parameter Name	Tag Number	Default Value	Header Parameter?	Reference
PatternWidth	106	1	Yes	A.2.1
PatternHeight	107	1	Yes	A.2.2
ComponentsPerSample	108	1	Yes	A.2.3
ImageFormat	84	N/A	Yes	A.2.4
MaxBitsPerComponent	102	12	Yes	A.2.5

Annex B Chunk Elements (Normative)

B.1 Small Chunk Elements

The set of small chunk elements defined in SMPTE ST 2073-1 Annex B.3.1 shall be extended by this standard to include the small chunk elements as defined by this standard in Table B.1. The small chunk elements shall be required or optional as specified by Table B.1.

Table B.1 – Small chunk elements that are defined by this standard

Small Chunk Element	Tag Number	Tag Type	Description
VendorSpecificData	0x4000	Optional	Tag that identifies a vendor-specific data defined in this standard
InversePermutation	0x4001	Required	Tag that identifies an inverse component permutation defined in this standard
InverseTransform	0x4002	Required	Tag that identifies an 8-bit inverse component transform as defined in this standard
InverseTransform16	0x4003	Required	Tag that identifies a 16-bit inverse component transform as defined in this standard
UniqueImageIdentifier	0x4004	Optional	Tag that identifies a unique image identifier as defined in this standard

B.2 Large Chunk Elements

This standard does not define any additions or changes to the large chunk elements defined in SMPTE ST 2073-1.

Annex C Colorimetry (Normative)

C.1 Colorimetry Specification

The image processing infrastructure that utilizes the VC-5 decoder, or external parameters (SMPTE ST 2073-1, Section 9.1), may provide the colorimetry information for interpreting the decoded component values. If this colorimetry information is not provided and ImageFormat equals 1 or 2, then the colorimetry shall be as specified in this annex. In all other cases the colorimetry is not defined by this standard.

C.2 Digital Representation

The R′, G′, B′, Y′, C'_1, and C'_2 components shall be the digital representation with precision n equal to the value of the **BitsPerComponent** parameter.

The R'_A, G'_A, B'_A, Y'_A, P'_1, and P'_2 components shall be the abstract representation.

C.3 Color Differences

The C'_1 color component corresponds to the gamma-corrected blue color difference component and the C'_2 color component corresponds to the gamma-corrected red color difference component.

C.4 Color Primaries

The color primaries shall be as listed in Table C.1.

Table C.1 – Conventional reference primaries and reference white

	CIE x	CIE y
Red primary	0.640	0.330
Green primary	0.300	0.600
Blue primary	0.150	0.060
Reference white	0.3127	0.3290

C.5 Gamma Correction

The exponent of the EOTF shall be 2.4 as specified by Rec. ITU-R BT.1886.

C.6 $Y'C'_1C'_2$ Color Matrix

The Y'_A component shall be computed as a weighted sum of the nonlinear R'_A, G'_A, and B'_A primary components using coefficients calculated from the reference primaries according to the method given in SMPTE RP 177:

$$Y'_A = 0.2126\,R'_A + 0.7152\,G'_A + 0.0722\,B'_A$$

The Y'_A component shall be scaled and offset to fit the integer range of the digital Y′ component according to the following equation:

$$Y' = \lfloor 219\, D\, Y'_A + 16\, D + 0.5 \rfloor$$

where

$D = 2^{n-8}$, and

n is the value of **BitsPerComponent**.

The color difference components shall be computed as follows:

$$P'_1 = \frac{0.5}{1 - 0.0722}(B'_A - Y'_A)$$

$$P'_2 = \frac{0.5}{1 - 0.2126}(R'_A - Y'_A)$$

The P'_1 and P'_2 color difference components shall be scaled and offset to fit the integer range of the digital C'_1 and C'_2 color difference components according to the following equations:

$$C'_1 = \lfloor 224\, D\, P'_1 + 128\, D + 0.5 \rfloor$$

$$C'_2 = \lfloor 224\, D\, P'_2 + 128\, D + 0.5 \rfloor$$

C.7 R′G′B′ Color Components

The R'_A, G'_A, and B'_A color components shall be scaled to the full range of the digital R′, G′, and B′ components, respectively, by the following equations:

$$R' = \lfloor 255\, D\, R'_A + 0.5 \rfloor$$

$$G' = \lfloor 255\, D\, G'_A + 0.5 \rfloor$$

$$B' = \lfloor 255\, D\, B'_A + 0.5 \rfloor$$

Note: The R′, G′, and B′ component values are full-range as described in SMPTE RP 2077 for n = 8, 10, and 12 bits.

Annex D Decoding Process (Informative)

The bitstream is decoded into an ordered set of component arrays as specified in SMPTE ST 2073-1. The components from a specific row and column in each of the component arrays form a component vector.

Referring to Figure D.1, after the bitstream is decoded into an ordered set of component arrays in channel order, the last stages of the decoding process and the image repacking process perform the following steps:

1. The inverse component transform is applied to the component vector at every row index and column index in the decoded component arrays (Section 8.5),

2. The inverse component permutation is applied to every component vector output by the inverse component transform (Section 8.6),

3. Every component vector output by the inverse component permutation is packed into a pattern element (Section 7.3).

4. The pattern elements are assembled into a sample array in order by the row index and column index of the component vectors from the ordered set of component arrays decoded from the bitstream.

The result is a sample array that contains component values in the same order and the same numerical range as originally presented to the encoder by the image unpacking process. For example, the bitstream representation of a Bayer image will be repacked into a Bayer image with the same component order and precision as the Bayer image that was originally presented to the image unpacking process prior to encoding.

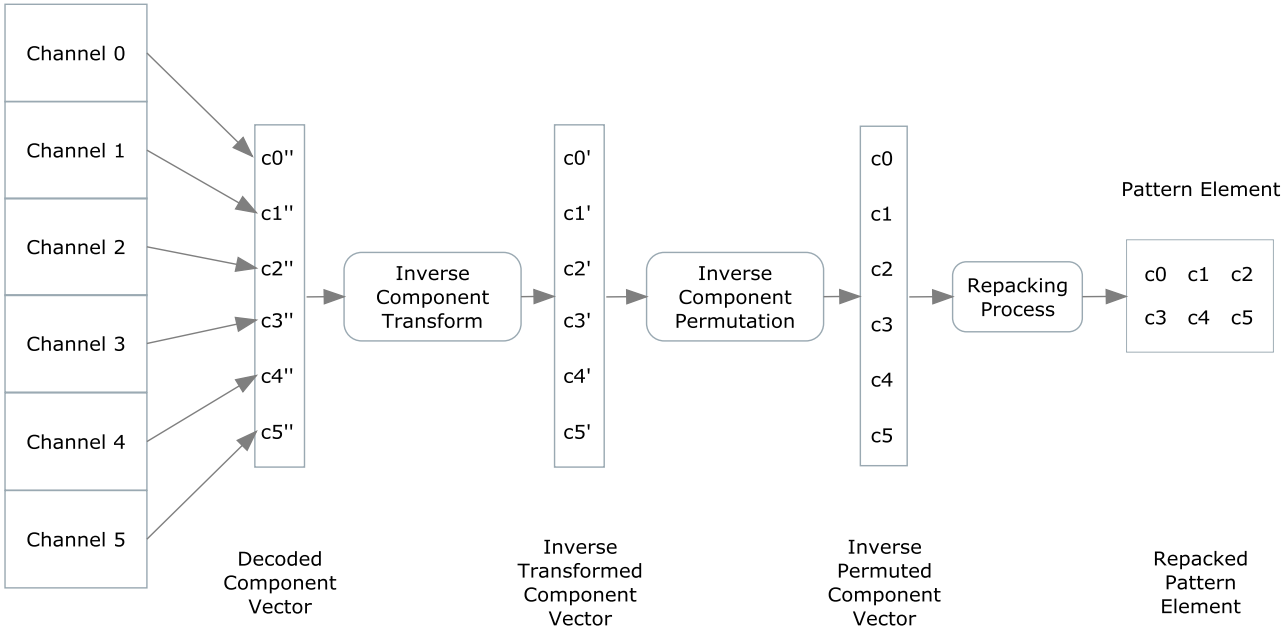

Figure D.1 – Inverse component transform, inverse component permutation, and repacking process

Annex E Encoding Process (Informative)

E.1 Unique Image Identifier

A unique image identifier (Section 11.1.5) associates an image with a sequence of images, such as a video clip or time-lapse photographs, and identifies the position of the image in the sequence. The use of the unique image identifier is out of scope.

E.2 Component Vector Order

For the purposes of the component transform and the component permutation a component vector is formed from the component values in all component arrays at the same row index and column index. The order of the component values in the component vector is the same as the order of the component arrays.

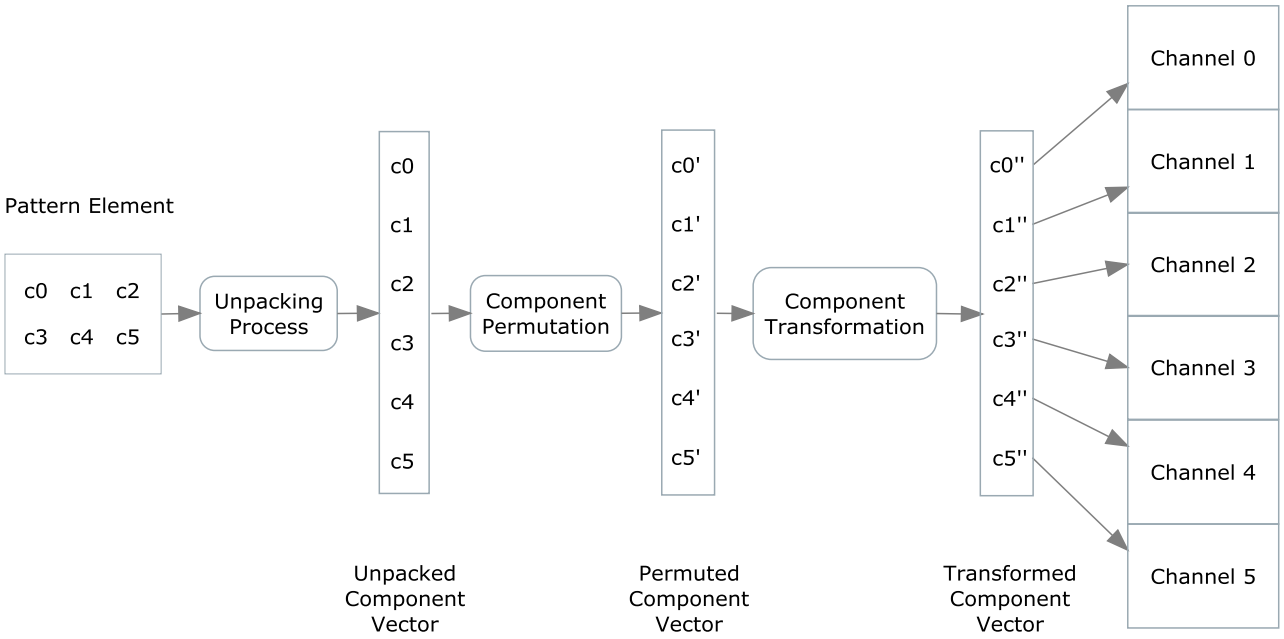

Figure E.1 – Unpacking process, component permutation, and component transform

E.3 Forward Component Permutation

The assignment of each component array present in the bitstream to a channel number is determined in part by the component permutation applied to the component arrays immediately after the image unpacking process (Figure E.1). The component permutation reorders the component arrays prior to application of the component transform. If the component permutation is not present in the bitstream, then the decoder will assume that the component permutation is the identity permutation.

E.4 Forward Component Transform

The component arrays present in the bitstream can be the result of a component transform applied to the component arrays immediately after the component permutation (Figure E.1).

Annex F Profiles and Levels

F.1 Profiles (Informative)

The tag-value pairs that are present in the bitstream header implicitly define the VC-5 profile. For example, the presence of the **ImageFormat** parameter in the bitstream header identifies the bitstream as having a profile that includes the capabilities of VC-5 Part 3.

F.2 Levels (Normative)

A level specifies a limit on the resources required by a decoder to decode a valid VC-5 bitstream. A decoder that conforms to a level shall be capable of decoding any valid bitstream with values for the codec state parameters listed in Table F.1 that are numerically less than or equal to the values for that level.

Table F.1 – Definition of codec levels

Level	ImageWidth	ImageHeight	MaxBitsPerComponent	ChannelCount
1.1	1920	1080	12	8
1.2	1920	1080	24	16
2.1	2048	1080	12	8
2.2	2048	1080	24	16
3.1	3840	2160	12	8
3.2	3840	2160	24	16
4.1	4096	2160	12	8
4.2	4096	2160	24	16
5.1	7680	4320	12	8
5.2	7680	4320	24	16
6.1	8192	4320	12	8
6.2	8192	4320	24	16
7.1	15360	8640	12	8
7.2	15360	8640	24	16
8.1	16384	8640	12	8
8.2	16384	8640	24	16

Note: The level is not explicitly represented in the bitstream.

Annex G Bibliography (Informative)

Note: All references in this document to other SMPTE documents use the current numbering style (e.g. SMPTE ST 274:2008) although, during a transitional phase, the document as published (printed or PDF) may bear an older designation (such as SMPTE 274M-2008). Documents with the same root number (e.g. 274) and publication year (e.g. 2008) are functionally identical.

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

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

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

Amendment 1:2012 to SMPTE ST 377-1:2011

Amendment 2:2012 to SMPTE ST 377-1:2011

SMPTE RP 177:1993, Derivation of Basic Television Color Equations

SMPTE RP 2077:2013, Full-Range Image Mapping

ISO 12234-2:2001, Electronic Still Picture Imaging, Removable Media, Part 2: TIFF/EP Image Data Format.

Recommendation ITU-R BT.709-5 (04/2002), Parameter Values for the HDTV Standards for Production and International Programme Exchange

Recommendation ITU-R BT.1886 (03/2011), Reference Electro-Optical Transfer Function for Flat Panel Displays used in HDTV Studio Production