SMPTE ST 2019-1:2016
Revision of
SMPTE ST 2019-1:2014

SMPTE STANDARD

VC-3 Picture Compression and Data Stream Format

Table of Contents

Approved
May 18, 2016

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 2019-1 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

The original scope of the VC-3 standard covered HD resolutions 1920 x 1080 (4:2:2 and RGB, 8 and 10 bit), and 1280 x 720 (4:2:2, 8-bit) as well as 1440 x 1080 and 960 x 720 (4.2:2, 8 bit) encoded according to ITU-R 709 as a constant bitrate codec. This backwards compatible revision adds five new, spatially resolution-independent, progressive-only compression IDs which also support 8-, 10- or 12-bit component depth, 4:4:4 and 4:2:0 YC_BC_R color subsampling, ITU Rec BT.2020 and provides for the optional inclusion of Alpha information (both loss-based and lossless for select compression IDs) and variable bitrate operation.

This revision of the standard also introduces the notion of a Profiles and Levels classification (Annex H). Encodings using compression IDs defined in prior revisions of this standard are collectively referred to as HD Profile. Encodings complying with this standard revision are referred to as RI (resolution independent) Profile.

The RI Profile is based on generalizations (removing or relaxing constraints in definitions) and extensions (backwards compatible additions) of some of the progressive HD scan types. Without applying the generalizations and extensions the bitstream formation of the RI profile matches the bitstream formation of the HD profile, except for header versions and compression IDs.

This revision of the VC-3 standard supports YC_BC_R and RGB video source data as follows:

YC_BC_R(A):

- Rasters:
  - Full raster (1920x1080, 1280x720)
  - Thin raster (1440x1080, 960x720), HD-only
  - RI (1x1 up to 16384x16384)
- Sampling:
  - HD
    - 4:2:2 at 8- and 10-bits
    - 4:4:4 at 10-bits
  - RI
    - 4:2:0(:4) at 8-, 10- and 12-bits
    - 4:2:2(:4) at 8-, 10- and 12-bits
    - 4:4:4(:4) at 10- and 12-bits
- Progressive
- Interlaced (HD-only)

RGB(A):

- Rasters:
  - Full raster (1920x1080)
  - RI (1x1 up to 16384x16384)
- Sampling:
  - HD: 10-bits
  - RI: 10- and 12-bits
- Progressive only

1 Scope

This standard specifies the compressed data format and decoding process for VC-3 compressed video data. VC-3 supports video up to 16384 x 16384 raster resolutions as YC_BC_R(A) (4:4:4(:4), 4:2:2(:4), 4:2:0(:4)) at 8-, 10- and 12-bit or as RGB(A) at 10- or 12-bit sampling depth.

VC-3 is an intra-frame compression format, independent of any predefined frame rate, which supports both constant (CBR) and variable (VBR) bitrate operations. The compression parameters are described by the contents of a header data region and attributes associated with a Compression ID described within this standard. The standard does not specify the encoding process nor does it define video frame rates although examples are used in this document.

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.

3 Normative References

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.

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

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

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 2036-1:2014, Ultra High Definition Television – Image Parameter Values for Program Production

Recommendation ITU-R BT.709-5 (04/02), Parameter Values for the HDTV Standards for Production and International Programme Exchange.

Recommendation ITU-R BT.2020-1 (06/2014), Parameter Values for Ultra-High Definition Television Systems for Production and International Program Exchange

4 Overview (Informative)

This section provides a brief overview of the VC-3 decoding process. Annex F System Overview is an informative annex where one can find a description of the system and encoding process for VC-3.

The decoding process is depicted in Figure 1. The VC-3 bitstream contains a header region followed by entropy codes, which encode the quantized Discrete Cosine Transform (DCT) coefficients for video and optional DCT-encoded or RLE-encoded alpha blocks. As shown in the figure, the header region of the VC-3 compressed bitstream is parsed to determine information about the compressed bitstream.

For Video data, entropy decoding is then performed, followed by inverse quantization. The inverse quantized coefficients are then transformed to the spatial domain via the Inverse Discrete Transform (IDCT), producing an initial output picture.

For DCT-based Alpha the decoding process follows the Video path.

For lossless Alpha data, the bitstream is decoded using a predictive, differential run-length encoding scheme, followed by zigzag reordering stage.

The VC-3 bitstream is intra-frame encoded, meaning that each compressed frame is independent of all other frames in the encoded bitstream.

The remainder of this standard describes the video sampling structure (Section 6), the compressed frame format (Section 7), and the decoding process (Section 8). In addition there are annexes that provide details regarding the entropy codeword format (Annex A), a suggested usage of the User Data Payload area (Annex B), the Compression ID definition tables (Annex C), the quantization tables (Annex D), the Variable Length Codeword (VLC) tables (Annex E), and compressed frame bitrates (Annex G).

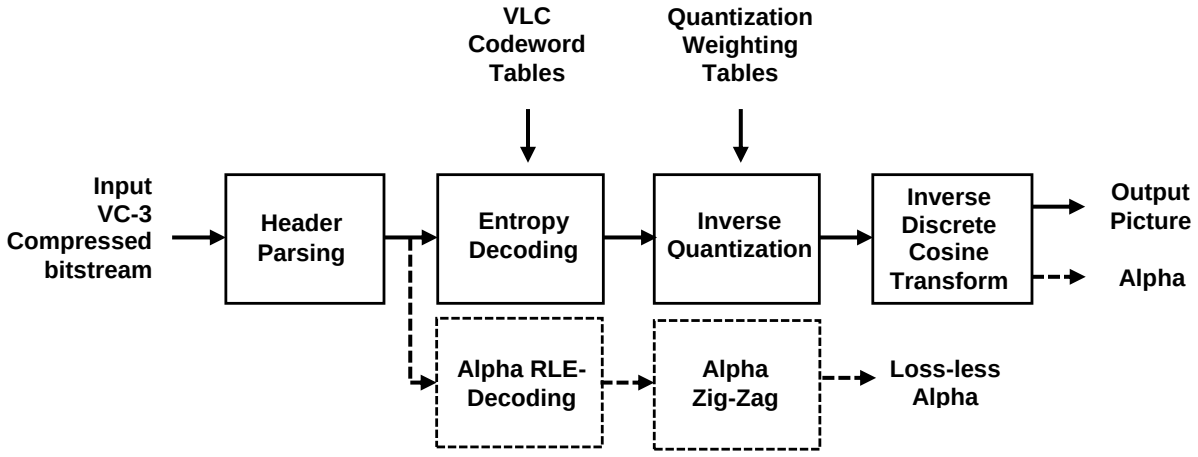

Figure 1 – Overview of the decompression process (dashed elements are optional)

5 Notation

The following notation is used in this document.

5.1 Values

d	Decimal value *d*
*x*b	Binary value *x*
0x*H*	Hexadecimal value *H*

5.2 Arithmetic Operators

+	Addition
−	Subtraction (as a binary operator) or negation (as a unary operator)
++	Increment
* or •	Multiplication
–	Division
$\|x\|$	Absolute value:

$|x| = x,$ when $x > 0$

$|x| = 0,$ when $x == 0$

$|x| = -x,$ when $x < 0$

$sgn(x)$	Sign:

$\mathrm{sgn}(x) = 1,$ when $x \geq 0$

$\mathrm{sgn}(x) = -1,$ when $x < 0$

$\lfloor x \rfloor$	Floor: Returns the largest integral value $\leq x$
$[x]_i$	Round to multiple: Returns the multiple of i closest to x

5.3 Logical Operators

TRUE/FALSE convention: This document uses the convention that a variable or expression evaluating to a nonzero value is equivalent to a condition that is TRUE and a variable or expression evaluating to a zero value is equivalent to a condition that is FALSE.

&&	logical AND
||	logical OR

5.4 Relational Operators

$>$	Greater than
$\geq$	Greater than or equal to
$<$	Less than
$\leq$	Less than or equal to
$==$	Equal to
$!=$	Not equal to

5.5 Bitwise Operators

A twos complement number representation is assumed where the bitwise operators are used.

>>	Shift right with sign extension.
<<	Shift left with zero fill.
&	Bitwise AND.

5.6 Assignment

$=$	Assignment operator.

5.7 Precedence Order of Operators

The precedence order of operators is defined as follows:

Operators	Type of operation	Associativity
$( )$	Expression	Left to Right
$++$	Postfix operator	Right to Left
$-$	Unary	Left to Right

* , /	Multiplicative	Left to Right
+ , –	Additive, Subtractive	Left to Right
<< >>	Shift	Left to Right
> , ≥ , < , ≤	Relational	Left to Right
==	Equality	Left to Right
&&, ||	AND, OR	Left to Right
=	Assignment operator	Right to Left

Operators are listed in descending order of precedence. If several operators appear in the same line, they have equal precedence. When several operators of equal precedence appear at the same level in an expression, evaluation proceeds according to the associativity of the operator either from right to left or from left to right.

5.8 Pseudocode Operations

The following operations are used in the pseudo code to define the decoding process.

• // is a comment to the line end

• A group of statements is treated functionally as a single (compound) statement if and only if it is enclosed in curly brackets { ... }.

• while (*condition*)
(*statement*)

specifies repeated execution of *statement* until condition is no longer true.

• for(*initial statement*; *condition*; *subsequent statement*)
(*primary statement*)

specifies evaluation of *initial statement* followed by evaluation of *condition*, and if *condition* is true, specifies repeated execution of *primary statement* followed by *subsequent statement* until *condition* is no longer true

• if (*condition*)
(*statement*)
else
(*alternate statement*)

statement is executed if *condition* is true, *alternate statement* is executed otherwise.

• switch (*discriminator*)
{
case (*value1)*[,(*value2*), (*value3*), (*value4*)]: (*statement1*); break;
case (*value5*)[,(*value6*), (*value7*), (*value8*)]: (*statement2*); break;
}

(*statement1*) is executed if (*discriminator*) matches any of the (*values*) following the case label.

5.9 Definition of Terminology

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

5.9.1 AC coefficient

Any DCT coefficient for which the frequency in one or both dimensions is non-zero.

5.9.2 alpha

The opacity of a pixel. A logical value of 0.0 denotes complete transparency, while a logic value of 1.0 denotes complete opacity.

5.9.3 amplitude

The absolute value of a DCT coefficient.

5.9.4 bitstream

An ordered series of bits that forms the coded representation of the data.

5.9.5 bitrate

The (bit) data rate at which the decoder consumes the bits of the input bitstream.

5.9.6 block

An 8 x 8 matrix of samples, or 64 equivalent transform coefficients.

5.9.7 byte-aligned

A bit in a coded bitstream is byte-aligned if its position is a multiple of 8 bits from the first bit in the stream.

5.9.8 byte

A sequence of 8 bits.

5.9.9 CBR

Constant bitrate encoding. The output data per time segment is constant.

5.9.10 coding unit

A self-contained VC-3 data segment. A coding unit is comprised of a Header, a Compressed Payload, and an End of Frame signature data section. Progressive frames require a single coding unit, while interlaced frames require two coding units, one for each field.

5.9.11 component

A matrix, block or single sample from one of the three matrices (one luma matrix and two color difference matrices or an RGB triplet) that make up a picture.

5.9.12 compressed payload

The section of data which contains the compressed data of a coding unit.

5.9.13 compression

Reduction in the number of bits used to represent an item of data.

5.9.14 compression ID

A unique identifier associated with a sample bit depth and encoding parameters such as variable length codeword tables and quantization weights. For HD bitstreams the Compression ID also implies the raster size and compressed frame size. This implication does not apply to RI-based bitstreams.

5.9.15 DC coefficient

The DCT coefficient for which the frequency is zero in both dimensions.

5.9.16 decoder

An embodiment of a decoding process.

5.9.17 decoding process

The process defined whereby a compressed bitstream is converted to a picture. The decoding process does not include the display rendering process, which can convert these samples to images of another colorspace, can apply format specific black and white levels, color primaries, matrix coefficients sample aspect ratios, etc., and can display the images with frequency and timing different from the sampled rate.

5.9.18 dequantization

The process of rescaling the quantized transform coefficients after their representation in the bitstream has been decoded and before they are presented to the inverse transform.

5.9.19 display process

The (non-normative) process by which reconstructed frames are displayed.

5.9.20 encoder

An embodiment of an encoding process.

5.9.21 encoding (process)

A process which reads a stream of input pictures and produces a valid coded bitstream.

5.9.22 flag

A variable which can take one of only two values is called a flag.

5.9.23 Frame

A frame contains lines of spatial information of a video signal. For progressive video, these lines contain samples starting from one time instant and continuing through successive lines to the bottom of the frame. For interlaced video, a frame consists of two fields.

5.9.24 field

For an interlaced video signal a "field" refers to the assembly of alternate lines. Therefore an interlaced frame is composed of two fields, a top field, containing the spatially top-most line of the image, and a bottom field, containing the spatially bottom-most line of the image. One of these fields will commence one field period later than the other.

5.9.25 frame rate

The rate at which frames are output from the decoding process.

5.9.26 HD raster

Rasters conforming to either 1920x1080p, 1920x1080i, 1280x720p (full raster), or 1440x1080p, 1440x1080i, 960x720p (thin raster).

5.9.27 header

A block of data in the coded bitstream containing the coded representation of a number of data elements pertaining to the coded data that follow the header in the bitstream.

5.9.28 macroblock

For 4:2:0 video data, the four 8 by 8 blocks of luma data and the two corresponding 8 by 8 blocks of color-difference data corresponding to a 16 by 16 section of the luma component of the picture, resulting in a total of 6 DCT blocks. For 4:2:2 video data, the four 8 by 8 blocks of luma data and the four corresponding 8 by 8 blocks of color-difference data corresponding to a 16 by 16 section of the luma component of the picture, resulting in a total of 8 DCT blocks. For (RGB) 4:4:4 data, the macroblock consists of a 16 by 16 section of sample data, resulting in a total of 12 DCT blocks.

5.9.29 macroblock scan line

A row of macroblocks running the entire width of the video raster.

5.9.30 parameter

A variable within the syntax which can take one of a range of values.

5.9.31 picture

Source, or decoded image data. A source or decoded picture consists of three rectangular matrices of 8-, 10- or 12-bit numbers representing the luma and two color-difference signals or representing the red, green and blue signals. See the relevant normative reference for the specific definition.

5.9.32 progressive

The property of frames where all the samples of the frame represent the same instance in time and lines are stored in consecutive order.

5.9.33 quantize, quantization

A process in which the continuous range of values of an input signal is divided into non-overlapping (but not necessarily equal) sub-ranges, and a discrete, unique value is assigned to each sub-range. A unique index is generated to represent this value.

5.9.34 raster

A picture with a particular width, height and frame layout.

5.9.35 resolution independent (RI) raster

A raster whose size is not predefined and shall be additionally specified. This term is used to differentiate from the pre-defined HD rasters.

5.9.36 run

The number of zero coefficients preceding a non-zero coefficient, in the scan order.

5.9.37 variable length coding (VLC)

A reversible procedure for coding, that assigns shorter code-words to symbols of higher probability and longer code-words to symbols of lower probability. The acronym VLC also indicates a variable length code.

5.9.38 VBR

Variable bitrate encoding. The amount of output data per time segment can vary.

5.9.39 Video Codec 3 (VC-3)

The name of the standard described here.

5.9.40 zigzag scanning order

A specific sequential ordering of the transform coefficients from (approximately) the lowest spatial frequency to the highest.

5.10 Acronym Definitions

The following acronyms are commonly used:

5.10.1 DCT
Discrete Cosine Transform

5.10.2 IDCT
Inverse Discrete Cosine Transform

5.10.3 LSB
Least Significant Bit.

5.10.4 MB
MacroBlock.

5.10.5 MSB
Most Significant Bit.

5.10.6 VLC
Variable Length Code.

5.10.7 VLD
Variable Length Decoding.

6 Video Sampling Structure

The VC-3 compression standard supports the following features:

6.1 HD Raster Constraints

The constraints in this section match the definitions of the previous revision (SMPTE ST 2019-1:2014) of this standard.

- 1080 line interlaced video
- 1080 line progressive video
- 720 line progressive video

For 1080 line Video raster sizes, the video sampling structure of YC_BC_R and RGB video shall be defined by SMPTE ST 274; for 720 line raster sizes, the video sampling structure of YC_BC_R video shall be defined by SMPTE ST 296.

For YC_BC_R video, the decoder's video output signal shall consist of Luminance (Y) and color-difference components (C_B and C_R) with a 4:2:2 (8-, 10--bit) or 4:4:4 (10-bit) sampling ratio. The component sample levels are tabulated in Table 1.

For RGB video, the decoder's output signal shall consist of 10-bit RGB components. The video sampling structure for RGB video shall always be 1920 samples per line and 1080 lines per frame (no thin raster support). The frame layout for RGB video shall be progressive. The component sample levels are tabulated in Table 2.

The decoder's video output signal for RGB video shall consist of Red, Green and Blue components. The RGB *encoder* may choose, on a macroblock basis, between encoding RGB or internally transcoded YC_BC_R values in the bitstream. Only the RGB format shall be returned by the *decoder.*

For HD rasters only CBR operation shall be permitted.

6.2 RI Raster Constraints

The constraints in this section apply to the additions made in this revision of this standard.

- Samples per line and lines per frame of the raster shall be no less than 1 and no more than 16384.
- For YC_BC_R 4:2:2 video the number of samples per line shall be even.
- For YC_BC_R 4:2:0 video the number of lines and the number of samples per line shall be even.
- Progressive video

The video sampling structure shall adhere to SMPTE ST 274 or SMPTE ST 2036-1 using VC3 in-band identification methods. The characteristics for other sampling structure mappings should be defined in an out-of-band bitstream container and will not be verified and ascertained by this decoder.

For YC_BC_R(A) video the decoder's video output signal shall consist of Luminance (Y) and color-difference components (C_B and C_R) with 4:2:0(:4) (8-, 10- or 12-bit), 4:2:2(:4) (8-, 10- or 12-bit) or 4:4:4(:4) (10- and 12-bit) sampling ratios respectively. The component sample levels are tabulated in Table 1.

For RGB(A) video the decoder's output signal shall either consist of 10- or 12-bit RGB(A) components. The component sample levels are tabulated in Table 2.

The decoder's video output signal for RGB(A) video shall consist of Red, Green and Blue components and an optional Alpha component. The RGB(A) *encoder* may choose between encoding the video as RGB (RGB encoding mode) or YC_BC_R (YC_BC_R encoding mode) values in the bitstream if the input is signaled to comply with Recommendation ITU-R BT.709 or Recommendation ITU-R BT.2020. Only the RGB format shall be returned by the *decoder*. The component sample levels for YC_BC_R encoding are tabulated in Table 2.

RI rasters may be encoded as constant or variable bitrate bitstreams. The operation mode shall be constant for the bitstream and shall be signaled in the Compressed Frame Header (see Section 7.2.2).

To avoid confusion and retain full backwards encoding compatibility, a VC-3 encoder shall use the HD profile encoding, whenever applicable, instead of the (otherwise identical) RI profile.

6.3 Frame Structure

Table 1 and Table 2 list the component sample levels for all encodings supported by this standard.

Table 1 – YC_BC_R Video: Sample levels for 8-, 10- and 12-bit source sampling bit-depth (Informative)

	Sampling bit depth		8-bit (0 … 255)	10-bit (0 … 1023)	12-bit (0 ... 4095)
Sample levels (BT 709 and BT 2020)	Y	Peak Range	1 … 254	4 … 1019	16 … 4079
		White Level	235	940	3760
		Black Level	16	64	256
		Number of levels between white and black (inclusive)	220	877	3505
	C_B, C_R	Gray Level	128	512	2048
		Excursion	+/- 112	+/- 448	+/- 1792
Sample levels (out-of-band)	Y	Peak Range	0 … 255	0 … 1023	0 … 4095
	C_B, C_R	Gray level	128	512	2048
		Excursion	+/- 127	+/- 511	+/- 2047
Alpha	A	Peak Range	0 … 255	0 … 1023	0 … 4095

Table 2 – RGB Video: Sample levels for 10- and 12-bit source sampling bit-depth (Informative)

	Sampling bit depth		10-bit (0…1023)	12-bit (0 … 4095)
Sample levels (RGB encoding mode)	R,G,B	Peak Range	4…1019	16 … 4079
		White Level	940	3760
		Black Level	64	256
		Number of levels between white and black (inclusive)	877	3505
Sample levels (YC_BC_R encoding mode)	Y	Peak Range	4…1019	16 … 4079
		While Level	940	3760
		Black Level	64	256
		Number of levels between while and black (inclusive)	877	3505
	C_B, C_R	Gray Level	512	2048
		Excursion	+/- 448	+/- 1792
Sample levels (out-of-band)	R,G,B	Peak Range	0 … 1023	0 … 4095
Alpha	A	Peak Range	0 … 1023	0 … 4095

As shown in Figure 2, the active video raster shall be subdivided into macroblocks, which are 16x16 blocks of spatially adjacent samples. A macroblock scan line shall be defined as a row of macroblocks

running the entire width of the video raster. The following variables are defined to specify the video raster subdivision into macroblocks:

N_W = the number of macroblocks per macroblock scan line; that is, the *width* of the macroblock scan line.

N_S = the number of macroblock scan lines in a frame or field.

N_T = the total number of macroblocks in a frame or field. Note that $N_T = N_W * N_S$.

l = macroblock index; $l = 0,1,2,...,N_T$, running through consecutive macroblock scan lines.

All rasters for which the number of active lines is not a multiple of 16 shall require additional (augmentation) lines to increase the total number of lines in the bitstream to the next multiple of 16 (Figure 2 below). During decoding the data contained in these augmentation lines shall be discarded.

RI rasters with a number of active samples per line not divisible by 16 shall require additional augmentation samples (padding) to increase the total number of samples per line in the bitstream to the next multiple of 16 (Figure 2 below). During decoding the data contained in these augmentation samples shall be discarded.

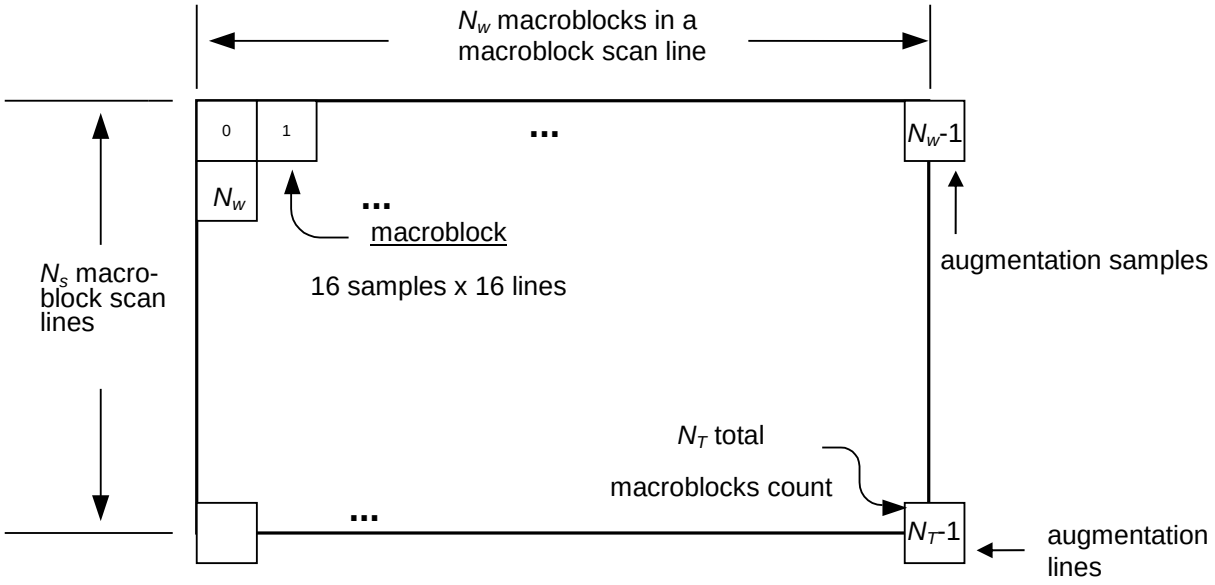

Figure 2 – Macroblock subdivision of the video raster

The macroblock subdivisions for the HD raster sizes supported by VC-3 shall be as shown in Figure 3 thru Figure 8 and tabulated in Table 3. These figures specify how to map individual macroblocks to a specific range of sample locations in the video raster.

As indicated in Table 3, three HD video rasters (bottom three rows) contain fewer samples per line than required by SMPTE ST 274 for 1080 line video rasters and SMPTE ST 296 for 720 line rasters. For these three cases, the video sampling structure shall be defined as a ¾ horizontally subsampled raster resulting in 1440 luma and 720 color-difference samples per line for 1080i and 1080p video and resulting in 960

luma and 480 color-difference samples per line for 720p video. These three sub-sampled rasters shall be denoted as 1080p - thin raster, 1080i - thin raster, and 720p - thin raster.

Note: Interlaced video can be partitioned on a field basis as required for field encoding (Figure 4), or on a frame basis as required for frame encoding (Figure 6).

Table 3 – Macroblock related parameters for the subdivision of the HD video raster

Source scan type	Lines per frame	Samples per line	Number of macroblocks per scan line: N_W	Number of macroblock scan lines per frame or field: N_S	Number of macroblocks per frame or field: N_T
Progressive	1080	1920	120	68 (frame)	8160 (frame)
Interlaced	1080	1920	120	34 (field)	4080 (field)
Progressive	720	1280	80	45 (frame)	3600 (frame)
Progressive	1080	1440	90	68 (frame)	6120 (frame)
Interlaced	1080	1440	90	68 (frame)[1)]	6120 (frame)[1)]
Progressive	720	960	60	45 (frame)	2700 (frame)

[1)] Note: Supported only for CID 1260; the MACF constraint to frame-based encoding for this CID is in Section 7.2.2.

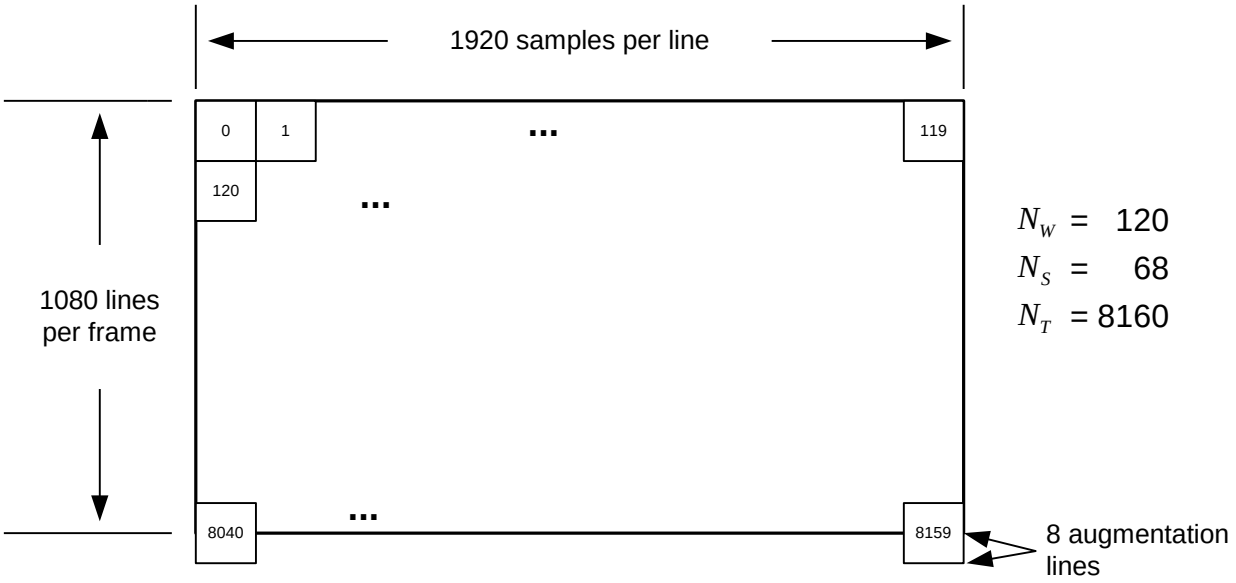

Figure 3 – 1080p raster subdivision into macroblocks

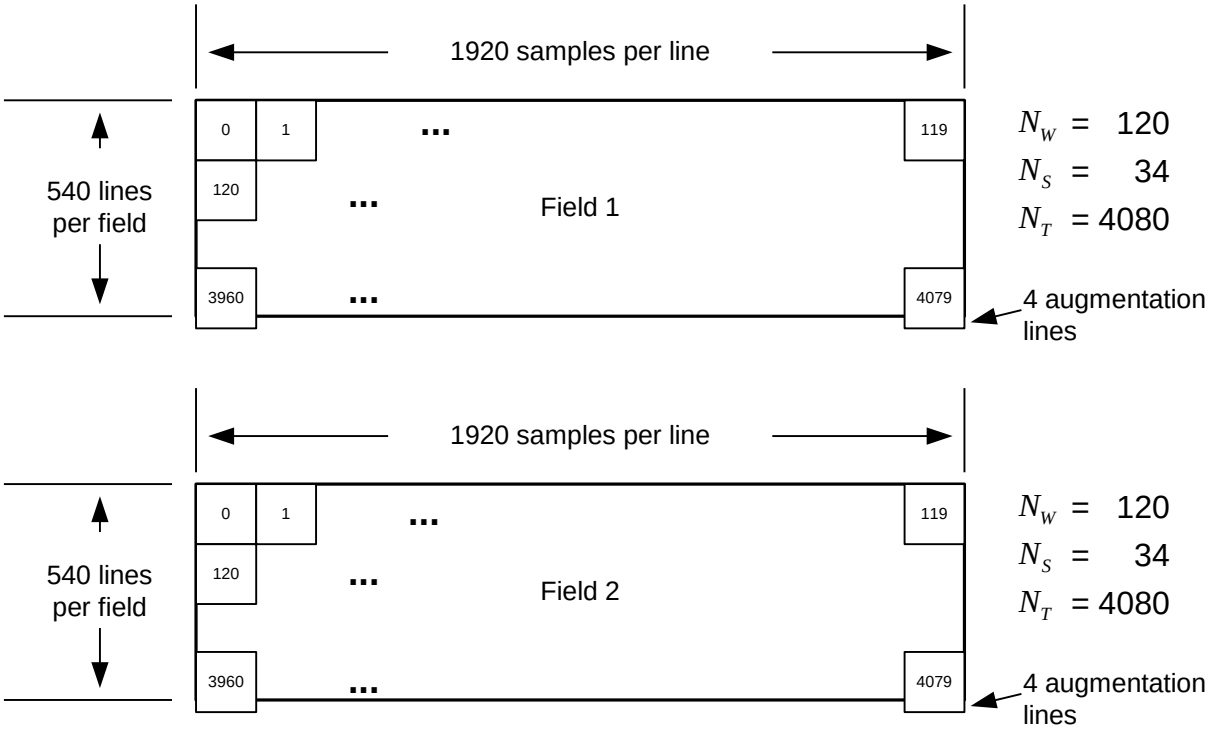

Figure 4 – 1080i raster sub-division into macroblocks

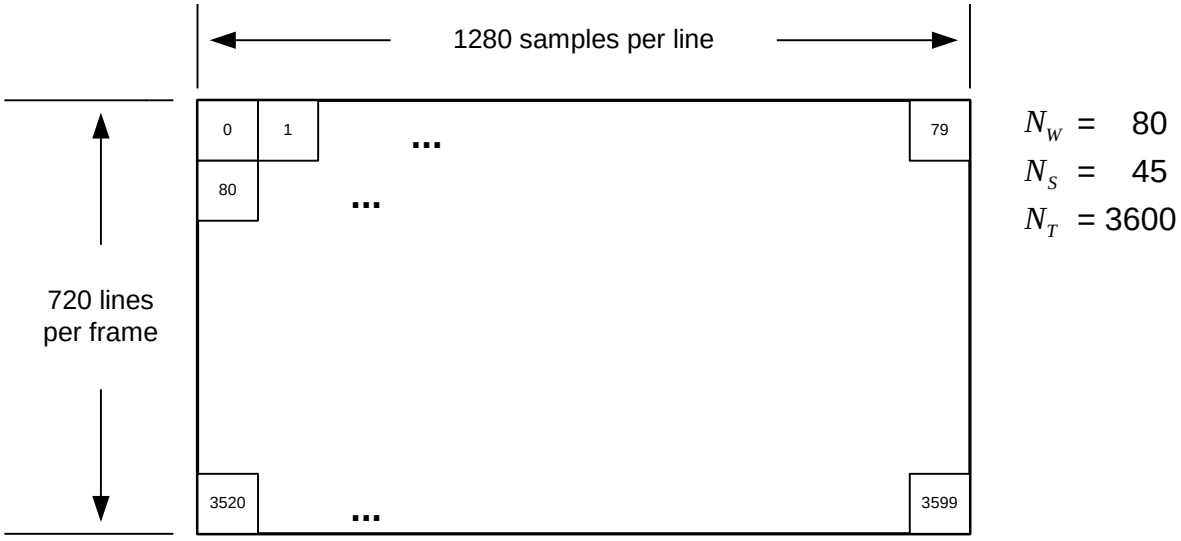

Figure 5 – 720p raster subdivision into macroblocks

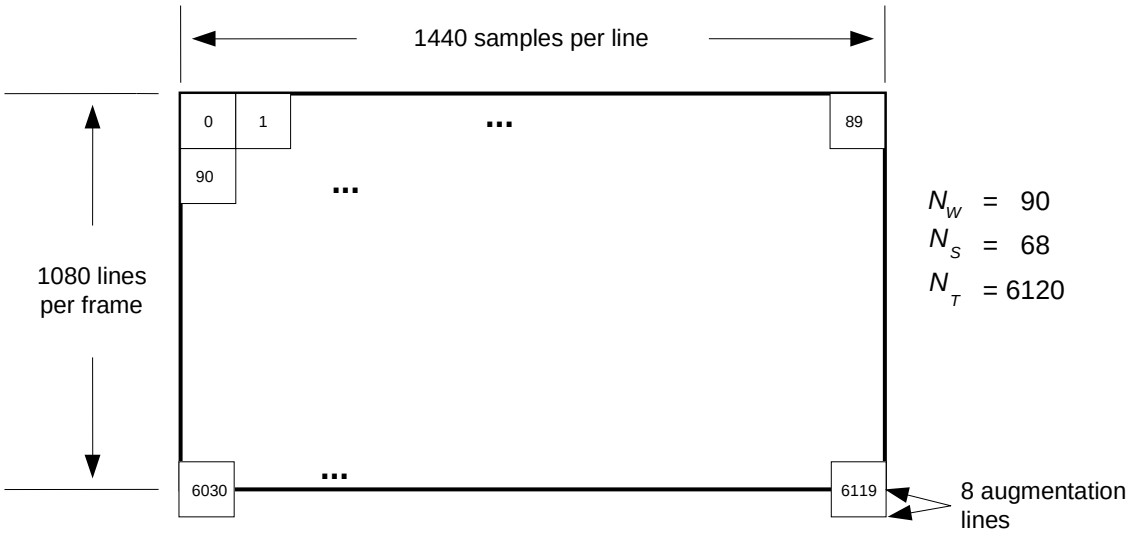

Figure 6 – 1080p – thin raster subdivision into macroblocks

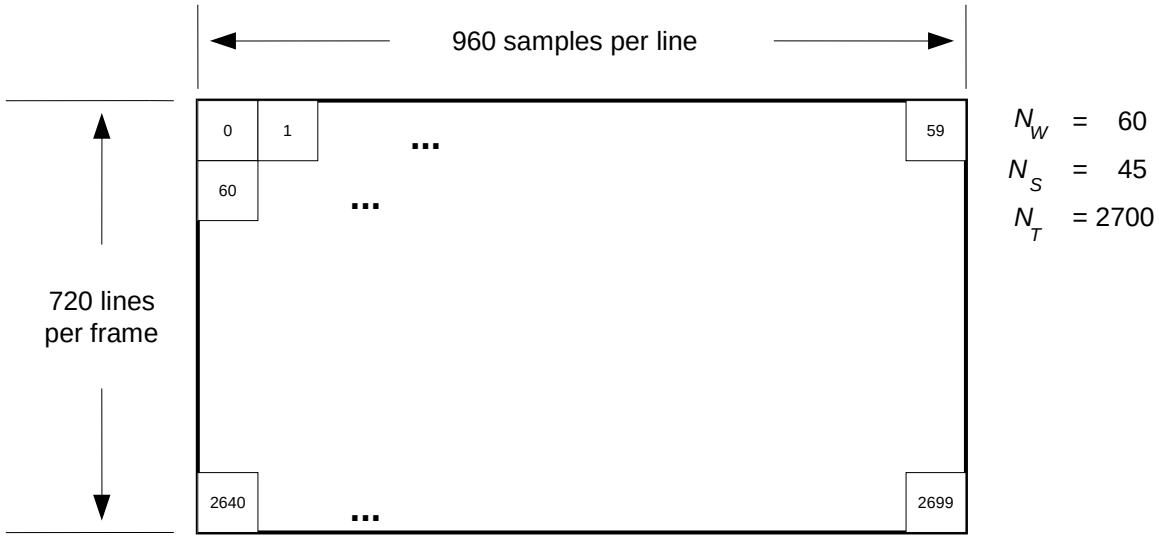

Figure 7 – 720p – thin raster subdivision into macroblocks

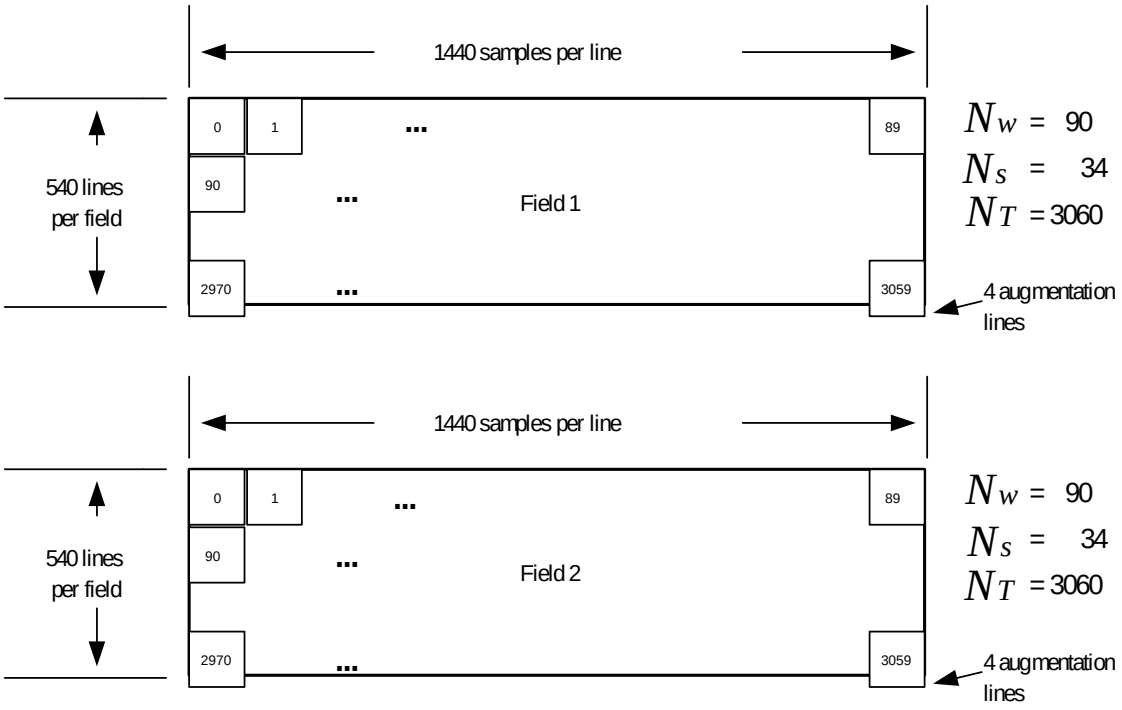

Figure 8 – 1080i – thin raster subdivision into macroblocks - field encoding

For RI rasters the macroblocks are organized in the same manner. The number of macroblocks in a row is the (augmented) raster width divided by 16, the number of macroblocks in a column is the (augmented) raster height divided by 16. The (optional) Alpha plane shall follow the definitions of the primary video.

For 4:2:0 (YC_BC_R) sampling, each macroblock shall be further subdivided into 8x8 pixel DCT blocks, as shown in Figure 9. Each DCT block shall consist of 8 pixels x 8 lines. Due to the 4:2:0 chroma sub-sampling ratio a DCT shall span 16 samples x 16 lines. The DCT blocks shall be ordered according to the indices listed in Table 4.

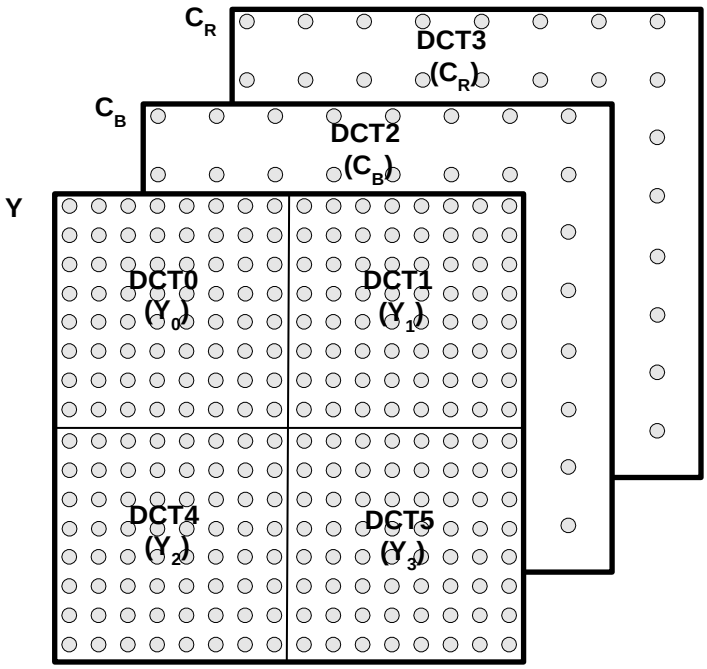

Figure 9 – Macroblock subdivision into 8x8 DCT blocks for YC_BC_R 4:2:0 sampling

Table 4 – DCT Block and DCT Index Correspondence for YC_BC_R 4:2:0 sampling

DCT Index k	DCT Block
0	Y_0
1	Y_1
2	C_{B0}
3	C_{R0}
4	Y_2
5	Y_3

For 4:2:2 (YC_BC_R) sampling, each macroblock shall be subdivided into 8x8 pixel DCT blocks, as shown in Figure 10. Each DCT block shall consist of 8 pixels x 8 lines. Due to the 4:2:2 chroma sub-sampling ratio a color-difference DCT block shall span 16 samples x 8 lines. The DCT blocks shall be ordered according to the indices listed in Table 5.

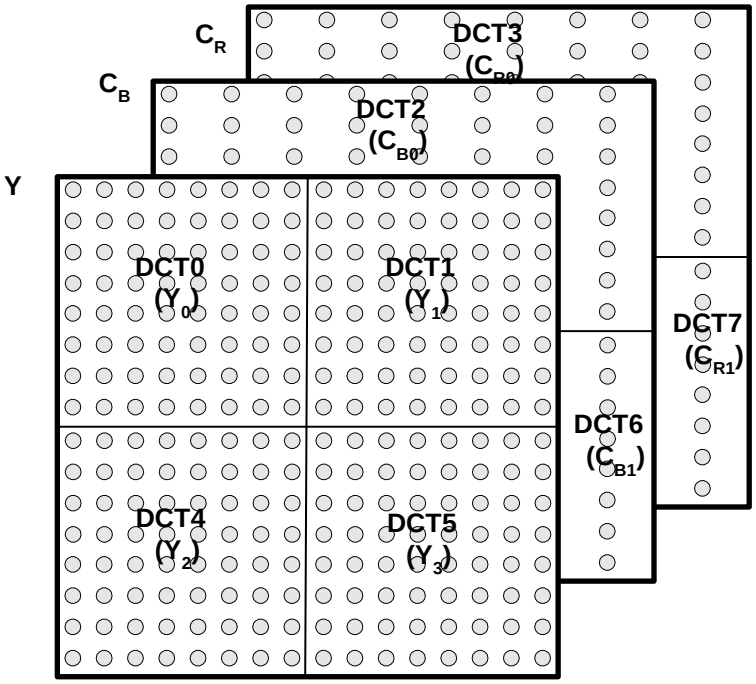

Figure 10 – Macroblock subdivision into 8x8 DCT blocks for YC_BC_R 4:2:2 sampling

Table 5 – DCT Block and DCT Index Correspondence for YC_BC_R 4:2:2 sampling

DCT Index k	DCT Block
0	Y_0
1	Y_1
2	C_{B0}
3	C_{R0}
4	Y_2
5	Y_3
6	C_{B1}
7	C_{R1}

For 4:4:4 YC_BC_R (RI) and RGB sampling, each macroblock shall be further subdivided into 8x8 pixel DCT blocks, as shown in Figure 11. Each DCT block shall consist of 8 pixels x 8 lines. The DCT blocks shall be ordered according to the indices listed in Table 6.

For RGB source sampling complying to ITU-R.BT 709 or ITU-R.BT 2020 the encoder can choose between encoding the original RGB values or equivalent YC_BC_R values on a macroblock basis: If the macroblock header bit-field ACF = 0b, the macroblock shall be encoded in *RGB Mode* and Ch1 shall be the red channel, Ch2 shall be the green channel, and Ch3 shall be the blue channel. If the macroblock header bit-field ACF = 1b, the macroblock shall be encoded in *YC_BC_R Mode* and Ch1 shall be the Luma channel, Ch2 shall be the C_B channel, and Ch3 shall be the C_R channel. See Section 7.3.1.1 for the definition of the ACF field.

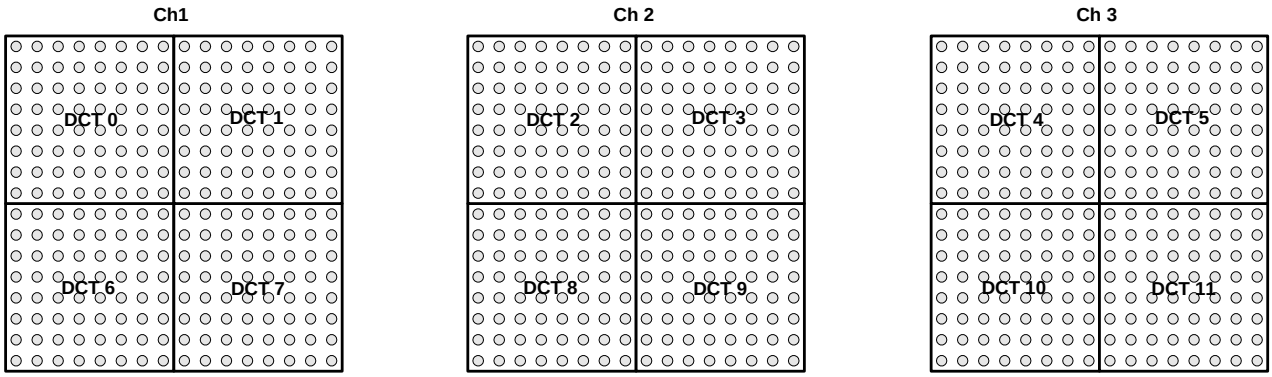

Figure 11 – Macroblock subdivision into 8x8 DCT blocks for 4:4:4 YC_BC_R and RGB sampling

Table 6 – DCT Block and DCT Index Correspondence for 4:4:4 YC_BC_R and RGB sampling

DCT Index k	DCT Block (RGB mode)	DCT Block (YC_BC_R mode)
0	R_0	Y_0
1	R_1	Y_1
2	G_0	C_{B0}
3	G_1	C_{B1}
4	B_0	C_{R0}
5	B_1	C_{R1}
6	R_2	Y_2
7	R_3	Y_3
8	G_2	C_{B2}
9	G_3	C_{B3}
10	B_2	C_{R2}
11	B_3	C_{R3}

The variable k denotes the DCT block index number, which ranges from 0 to 7. Figure 12 depicts the 8x8 block of video samples, $x_k^l(i, j)$ which comprises a DCT block. The index i and line index j range from 0 to 7.

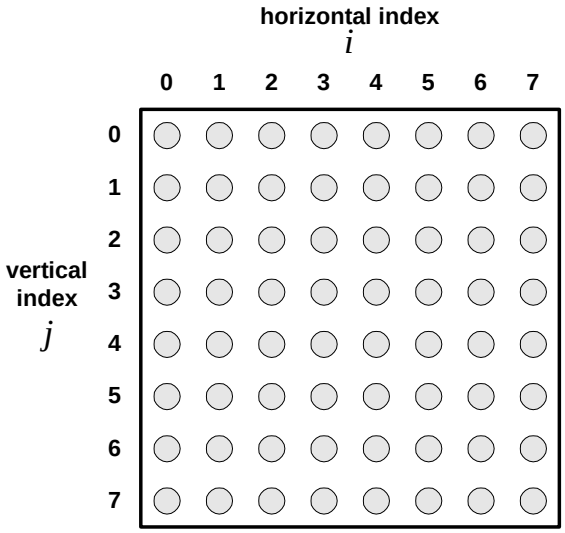

Figure 12 – $x_k^l(i, j)$: A block of 8x8 video sample data

For Compression ID 1260, two different macroblock modes are permissible. In Frame Macroblock Mode, the macroblock shall be partitioned into 8x8 DCT blocks, where each DCT block is comprised of data from both fields. This is depicted in Figure 13 where the top field is indicated by the lines with solid fill, and the bottom field is indicated by lines with no fill. In Field Macroblock Mode, depicted in Figure 14, the DCT blocks shall be comprised of only lines from a single field either top or bottom.

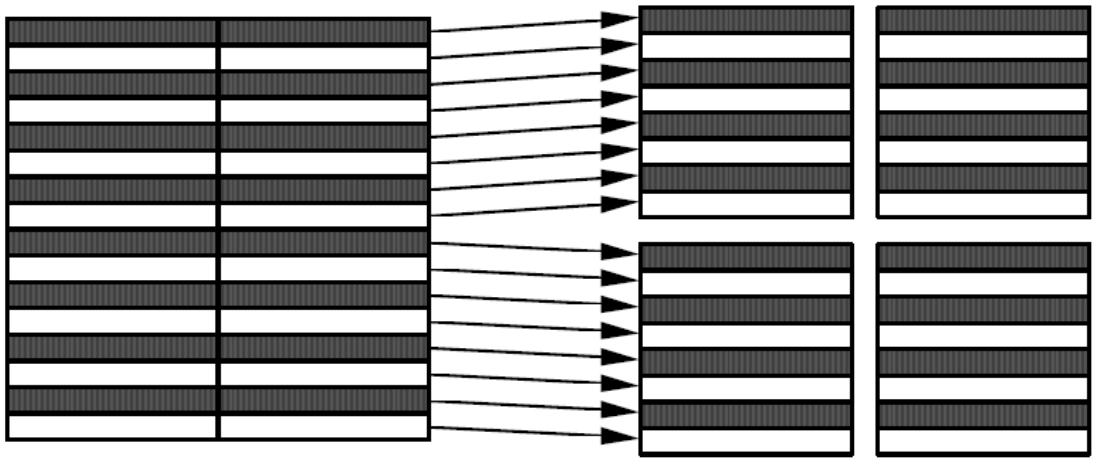

Figure 13 – Frame Macroblock Mode

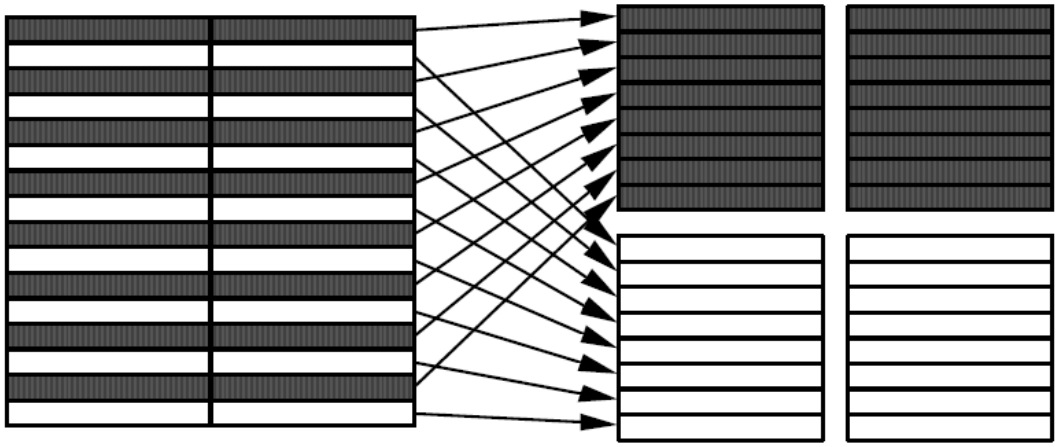

Figure 14 – Field Macroblock Mode

Alpha is broken down into analogous Alpha macroblocks sized 16 samples x 16 lines, matching the location of the macroblocks of the primary video.

For DCT-based alpha compression these macroblock shall be further subdivided into 4 DCT blocks, as shown in Figure 15. Each DCT block shall consist of 8 pixels x 8 lines. The DCT blocks for the Alpha channel shall be ordered as listed in Table 7 and shall use the same quantization matrix as defined for the Y component of the compression ID.

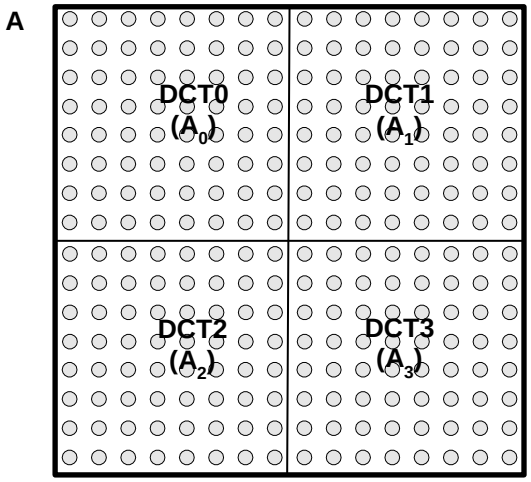

Figure 15 – Macroblock subdivision into 8x8 DCT blocks for Alpha sampling

Table 7 – DCT Block and DCT Index Correspondence for Alpha sampling

DCT Index k	DCT Block
0	A_0
1	A_1
2	A_2
3	A_3

For lossless differential RLE-compression the Alpha block shall be scanned as shown in Figure 16 (Alpha Zig-Zag).

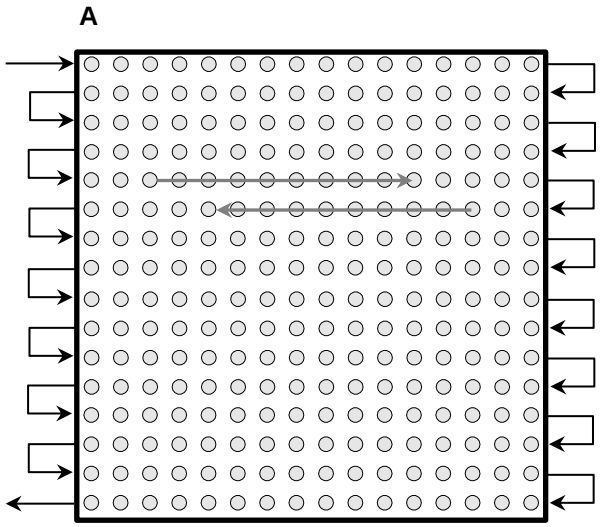

Figure 16 – Zig-zag scan of 16x16 Alpha blocks (Differential RLE encoding)

7 Compressed Frame Format

A VC-3 compressed bitstream shall be comprised of continuously concatenated coding units (Figure 17). A coding unit shall consist of a Header, a Compressed Payload, and an End-Of-Frame (EOF) signature of 4 bytes. Table C.1 and Table C.2 specify the Compressed Frame, Coding Unit, and Compressed Payload sizes for the range of VC-3 bitstreams with the corresponding Compression IDs.

The Header shall be as described in Section 7.2. It shall contain parameters required to decode Compressed Data contained in the Compressed Payload, which is described in Section 7.3. The Compressed Payload shall contain the Coding Unit's compressed data, see Section 7.3.1. If the length of the Compressed Data is less than the Compressed Payload Size for CBR encoding, the remainder of the Compressed Payload Area shall be filled with padding bytes as detailed in Section 7.3.2. The padding bytes shall be stored in the Compressed Payload Padding area. This padding step shall be omitted for VBR coding.

The compressed data for a single video frame shall require either one or two coding units:

- Progressive video shall require one coding unit and be encoded using frame encoding.
- Interlaced video shall require either one coding unit (frame encoding) or two coding units (field encoding, see Figure 18).

In the case of frame encoding, a single coding unit shall contain the compressed data representing all the video data of a single frame. In the case of the field encoding, two coding units shall hold the compressed data for a single frame. The first coding unit shall contain the compressed representation of the first video field, while the second coding unit shall contain the representation of the second video field. The second coding unit shall be concatenated onto the end of the first to create a compressed frame.

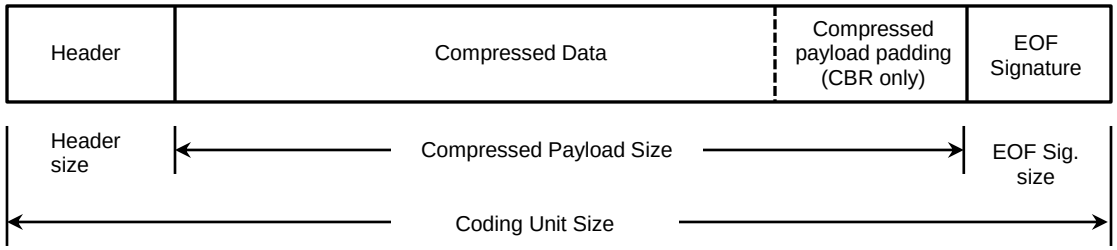

Figure 17 – Coding unit data structure

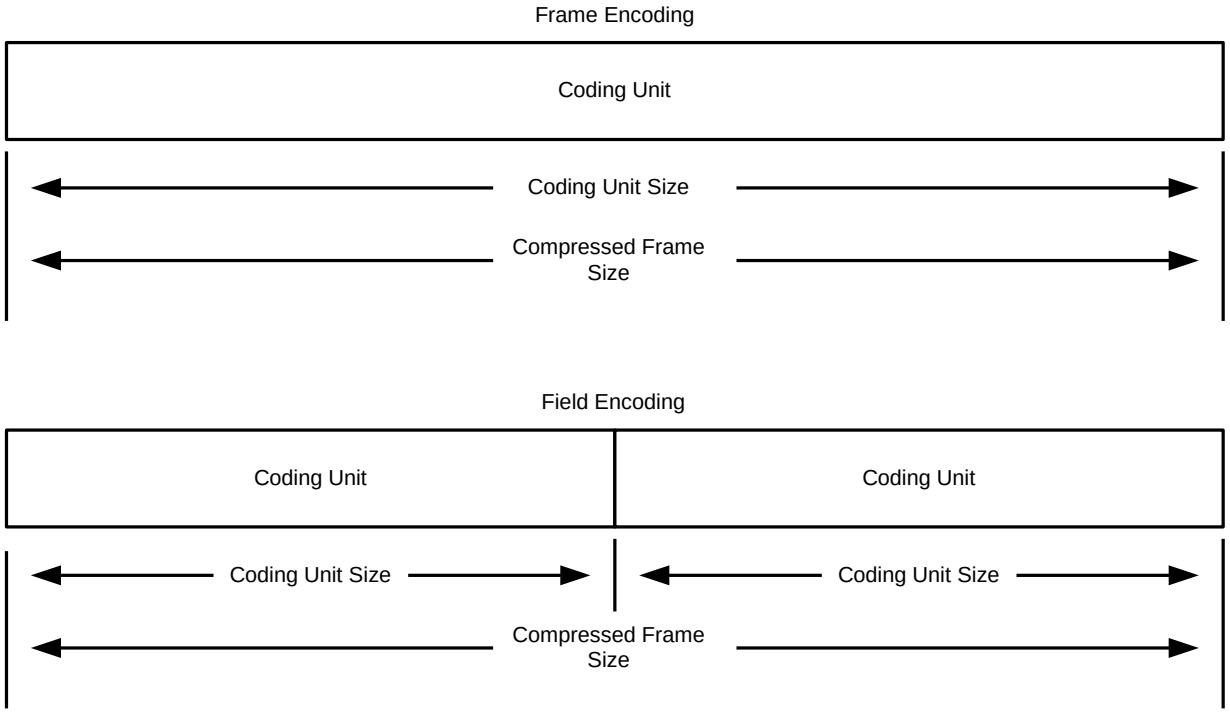

Figure 18 – Compressed frame data structure

7.1 Compressed Frame Size

For HD rasters the number of compressed bytes per frame C_{ref} for a particular Compression ID shall be as specified in Table C.1. HD rasters can only be compressed in CBR mode.

For RI rasters and CBR encoding C_{ref} shall be calculated according to

$$C_{ref}(f_A) = \left[(1 + f_A) \cdot \left(C_0 * \frac{N_T}{M_T}\right)\right]_{4096} \quad , \min\left(C_{ref}\right) = 8192 \qquad \textbf{(7.1)}$$

where C_0 and M_T denote the reference frame size and macroblock count for the Compression ID, taken from Table C.2, N_T is the total number of macroblocks calculated according to Section 6 and f_A = 0.5 is a constant, encoding-independent augmentation factor if alpha is present. The minimum number accepted for C_{ref} is 8192.

For RI rasters and VBR encoding C_{ref} shall denote the maximum compressed frame size permitted, except if lossless alpha compression is used. For this case, C_{ref} (0) shall denote the maximum compressed frame size of non-Alpha components, to which the variable size alpha encoding is then added.

Note 1: The following pseudo code calculates the compressed frame size of an RI raster of dimensions *width* and *height*, according to (7.1).

```
RIsize(width, height, f_A, C_0)
{
  N_W = width/16;
  if (width % 16 != 0)
    N_W++;                    // Padding required

  N_S = height/16;
  if (height % 16)
    N_S++;                    // Padding required

  size = (1+f_A)*C_0*(N_W*N_S)/8160;
  rem  = size % 4096;

  if (rem >= 2048)            // Round-up limit
    size += 4096-rem;         // Round UP
  else
    size -= rem;              // Round DOWN

  if (size < 8192)            // Lower size limit
    size = 8192;

  return size;
}
```

Note 2: The video frame rate at which the compression is applied will determine the compressed video *bitrate*. For example, the Compressed Frame Size of Compression ID 1241 is 917504 bytes. At a video frame rate of 29.97 frames per second, the compressed bitrate is 8*29.97*917504 = 219.98 Mbps. The informative Annex G, “Compressed Bitrates” provides examples of bitrates as a function of Compression ID and video frame rate.

Note 3: For lossless Alpha/VBR the maximum compressed frame size is approximately given by

$$C_{maxVBR} \approx C_{ref}(0) + N_T \cdot C_A$$

where C_A is the size of an Alpha macroblock stored as packed raw according to Section 7.3.1.2.

7.2 Header Definition

Header size shall depend on Compression ID. For HD rasters the contents of the 640 byte Header area shall be as summarized in Table 8. For RI rasters the header size shall depend on the raster height. If the raster height is less or equal 1088, the header size shall be 640 bytes. For each additional 16 lines (whole or partial), 4 bytes shall be added to the header size, which shall be recorded in the header prefix.

The following subsections define the header data regions. The byte offsets shown in Table 8 and the figures of this section shall be relative to the start of the coding unit. The content of the reserved areas shall be 0b. Decoder implementations should not process these regions.

Table 8 – Content of header data regions

Starting Offset	Data Region Length (bytes)	Header Data Region Contents	Subsection
0x000	0x005	Header Prefix	7.2.1
0x005	0x003	Coding Control A	7.2.2
0x008	0x010	Reserved	–
0x018	0x00B	Image Geometry	7.2.3
0x023	0x005	Reserved	–
0x028	0x004	Compression ID	7.2.4
0x02C	0x001	Coding Control B	7.2.5
0x02D	0x003	Reserved	–
0x030	0x001	Time Code Control	7.2.6
0x031	0x008	Time Code	7.2.7
0x039	0x026	Reserved	–
0x05F	0x001	User Data Control	7.2.8
0x060	0x104	User Data Payload	7.2.9
0x164	0x003	Reserved	–
0x167	0x009	Macroblock Scan Indices Control	7.2.10
0x170	At least 0x110	Macroblock Scan Indices Payload	7.2.11

7.2.1 Header Prefix

The header prefix shall consist of the five bytes given in Table 9.

Table 9 – Header Prefix data area contents

Byte Offset	Contents
0x000	HS_0
0x001	HS_1
0x002	HS_2
0x003	HS_3
0x004	HVN

HS: Header Size. The value of HS shall depend on values of Compression ID, samples per line and active lines per frame.

$HS = (HS_0 << 24) + (HS_1 << 16) + (HS_2 << 8) + HS_3$

HVN: Header Version Number. The value of HVN shall be dependent upon Compression ID as follows

0x01: Header version 1: Compression ID1235, 1237, 1238, 1241, 1242, 1243, 1244, 1250, 1251, 1252, 1253

0x02: Header version 2: Compression ID 1256, 1258, 1259, 1260

0x03: Header version 3: Compression ID 1270, 1271, 1272, 1273, 1274

For HVN values 0x01 and 0x02 the values of HS_0, HS_1, HS_2, and HS3 shall be 0x00, 0x00, 0x02, and 0x80 respectively.

HVN values 0x01 and 0x02 are HD profile.
HVN value 0x03 is RI profile.

Note: An encoder creating e.g. a Compression ID 1235 bitstream can write HVN=0x01 into the bitstream even when implemented according to this version of the standard. The HVN signals compliance of the bitstream to the definitions of specific Compression IDs, not to a specific version of this standard.

7.2.2 Coding Control A

Coding Control A area shall hold the following data fields as shown in Figure 19.

Byte Offset	**msb**							**lsb**
0x005	0	0	0	VBR	0	0	FFC_1	FFC_0
0x006	1	0	MACF	CRCF	0	0	0	0
0x007	1	0	1	0	0	PMA	LLA	ALP

Figure 19 – Coding Control A data area

VBR: Variable Bitrate Encoding Flag

The value of the VBR flag shall be constant throughout the bitstream and signal the absence of the Compressed Payload Padding in the bitstream.

0b: Constant bitrate encoding (CBR).
1b: Variable bitrate encoding (VBR). Only CID 1270-1274.

FFC: Field/Frame Count

01b: Progressive frame or interlaced frame in frame encoded mode.
10b: Field 1 of interlaced frame in field encoded mode.
11b: Field 2 of interlaced frame in field encoded mode.

MACF: Macroblock Adaptive Control Flag:

The MACF bit-field shall control the indication of adaptive Macroblock Field/Frame selection for macroblocks.

0b: Macroblocks do not use field macroblock mode
1b: Macroblocks may use either field or frame macroblock mode

MACF == 1 b shall always occur for CID 1260 and shall be 0 b for all other Compression IDs.

CRCF: CRC flag

0b: CRC shall not be present in EOF signature
1b: CRC shall be present in EOF signature

ALP: Alpha flag (RI-only, shall be 0b for HD rasters)

0b: Alpha channel not present

1b: Alpha channel present; only CID=1270, 1271, 1272 and 1273

LLA: Lossless Alpha flag (only CID=1270; shall be 0b for all others)

0b: Alpha, if present, is encoded using DCT-based compression

1b: Alpha, if present, is encoded using differential RLE compression.

PMA: Pre-multiplied Alpha (RI only; shall be 0b for HD rasters)

0b: Alpha has not been applied to video channels
1b: Alpha has been applied to the video channels prior to encoding.
Requires out-of-band encoding (CLV=11b, see Section 7.2.5, Coding Control B).

7.2.3 Image Geometry

The Image Geometry area shall hold the following data fields as shown in Figure 20.

Byte Offset	msb							lsb
0x018	$ALPF_1$							
0x019	$ALPF_0$							
0x01A	SPL_1							
0x01B	SPL_0							
0x01C	0	0	0	0	$PARC_1$		$PARN_1$	
0x01D	NAL_1							
0x01E	NAL_0							
0x01F	$PARC_0$							
0x020	$PARN_0$							
0x021	SBD			1	1	0	0	0
0x022	1	0	0	0	1	SST	0	0

Figure 20 – Image Geometry data area

ALPF: For HD rasters the Active Lines Per Frame value shall depend on the compression ID. The range of permissible values shall be as listed in the “Active lines” column of Table C.1.

For RI rasters the active lines-per-frame value shall not depend on a compression ID. The value of active lines-per-frame shall be no less than 1 and no more than 16384.

$ALPF = (ALPF_1 << 8) + ALPF_0$.

SPL: For HD rasters the samples-per-line value shall depend on the compression ID. The range of permissible values shall be as listed in the “Samples per line” column of Table C.1.

For RI the samples-per-line value shall not depend on a compression ID.

The value of samples-per-line shall be no less than 1 and no more than 16384.

If the sample-per-line value is not divisible by 16 each line shall be sample-padded in the encoded bitstream to the next multiple of 16.

$SPL = (SPL_1 << 8) + SPL_0$.

NAL: Number of Active Lines shall have the same value as ALPF.

$NAL = (NAL_1 << 8) + NAL_0$.

PAR: Pixel Aspect Ratio, expressed as a rational

$$a_p = \frac{\Delta x}{\Delta y} = \frac{\mathrm{PAR_C}}{\mathrm{PAR_N}}$$

$PAR_C = (PARC_1 << 8) + PARC_0$
$PAR_N = (PARN_1 << 8) + PARN_0$

HD rasters: $PARC_1 = PARC_0 = PARN_1 = PARN_0 = 0$. The Pixel Aspect Ratio is defined implicitly by the CID definition.

RI rasters The combination $PAR_C = PAR_N = 0$ (discriminatory value) is reserved. It signals that the pixel aspect ratio is not described in the bitstream (out-of-band definition).

For non-discriminatory values PAR_C and PAR_N shall both be non-zero and the effective value of a_p shall be in the range] 0.5, 2 [.

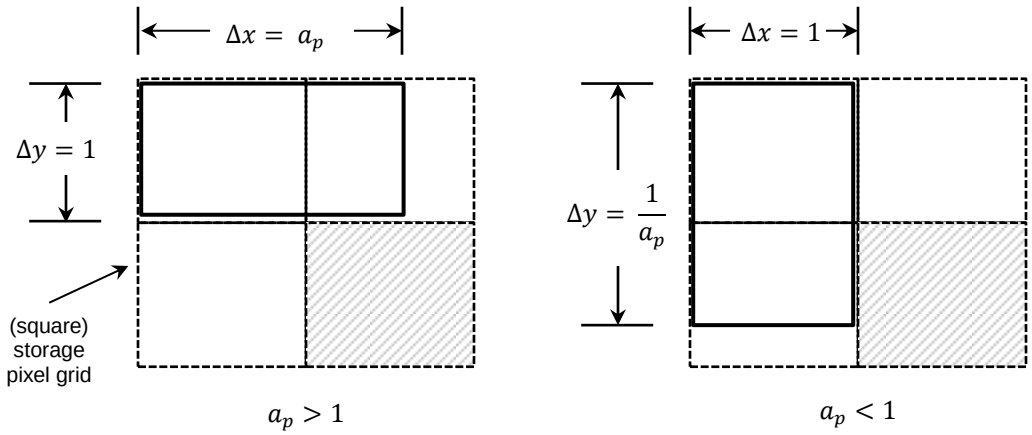

Figure 21 – Display pixel stretching according to a_p

SBD: Sample bit depth – shall correspond to the “Sample bit depth” column of Table C.1 or C.2.

011b: 12-bits per sample (RI only)
010b: 10-bits per sample
001b: 8-bits per sample

SST: Source scan type – shall correspond to “Source scan type” column of Table C.1 or C.2.

0b: Progressive scan
1b: Interlaced scan

7.2.4 Compression ID

The Compression ID area is depicted in Figure 22. This four-byte area shall hold a binary representation of the Compression ID values listed in the “Compression ID” column of Table C.1.

Byte Offset	msb						lsb
0x028	CID_3						
0x029	CID_2						
0x02A	CID_1						
0x02B	CID_0						

Figure 22 – Compression ID data area

Compression ID Example Compression ID 1250 has a hexadecimal representation of 0x000004E2. In this case CID_3 = 0x00,, CID_2= 0x00, CID_1 = 0x04, and CID_0 = 0xE2.

7.2.5 Coding Control B

The Coding Control B data area fields shall be as shown in Figure 23.

Byte Offset	msb						lsb
0x02C	FFE	SSC		0	0	CLV	CLF

Figure 23 – Coding Control B data area

It shall contain the following fields:

FFE: Field or Frame Encoding

0b: Field Encoding

1b: Frame Encoding

If SST == 0b (progressive)
then FFE = 1b (frame encoding)

If SST == 1b (interlaced) and Compression ID != 1260
then FFE = 0b (field encoding)

If SST == 1b (interlaced) and Compression ID == 1260
then FFE = 1b (frame encoding)

SSC: Sub Sampling Control

00b: 4:2:2 Sub Sampling

01b: 4:2:0 Sub Sampling – valid only for HVN > 2

10b: 4:4:4 Sub Sampling – valid only for Compression IDs 1256, 1270.

CLF: Color Format

0b: Bitstream is encoded using the YC_BC_R format rules and tables

1b: Bitstream is encoded using the RGB format rules and tables – only Compression IDs 1256, 1270

CLV: Color Volume

00b: Recommendation ITU-R BT.709, Part 2, Section 3: D'_R, D'_G, D'_B, D'_Y, D'_{CB}, D'_{CR}

01b: Recommendation ITU-R BT.2020, Table 5: D'_R, D'_G, D'_B, D'_Y, D'_B, D'_R

– Non-constant Luma mapping (709 compatible).

Valid only for Compression IDs 1270-1274

10b: Recommendation ITU-R BT.2020, Table 5: D'_{YC}, D'_{BC}, D'_{RC}

C – Constant Luma mapping.

Valid only for Compression IDs 1270-1274

11b: Out-of-band: $\mathcal{D}_R$, $\mathcal{D}_G$, $\mathcal{D}_B$, $\mathcal{D}_Y$, $\mathcal{D}_{CB}$, $\mathcal{D}_{CR}$ digital component representations as defined in the out-of-band mapping document.

– Color volume described out-of-band.

Disables internal color conversion for RGB encodings

Note: Be aware that Recommendation ITU-R BT.2020 defines RGB only for Non-constant Luma mapping (gamma-corrected RGB input). Therefore CLF=1b (RGB) cannot be paired with CLV=10b (Recommendation ITU-R BT.2020 Constant Luma mapping).

7.2.6 Time Code Control

The Time Code Control data area shall be as shown in Figure 24.

Byte Offset	msb							Lsb
0x030	TCP	0	0	0	0	0	0	0

Figure 24 – Time Code Control data area

TCP: Time Code Present

0b: Time code not present
1b: Time code present

7.2.7 Time Code Data Area

If the TCP flag of Section 7.2.6 is set, then the 8 bytes of the Time Code data area shall be filled with time code conforming to SMPTE ST 12-1 as specified in Table 10.

.

Table 10 – Time Code data area

Byte\bit	7	6	5	4	3	2	1	0
0x031	Binary Group 1 - BG1				Units of Frames			
	3	2	1	0	8	4	2	1
0x032	Binary Group 2 - BG2				Tens of Frames			
	3	2	1	0	Color Frame	Drop Frame	20	10
0x033	Binary Group 3 - BG3				Units of Seconds			
	3	2	1	0	8	4	2	1
0x034	Binary Group 4 - BG4				Tens of Seconds			
	3	2	1	0	Field ID	40	20	10
0x035	Binary Group 5 - BG5				Units of Minutes			
	3	2	1	0	8	4	2	1
0x036	Binary Group 6 - BG6				Tens of Minutes			
	3	2	1	0	X	40	20	10
0x037	Binary Group 7 - BG7				Units of Hours			
	3	2	1	0	8	4	2	1
0x38	Binary Group 8 - BG8				Tens of Hours			
	3	2	1	0	X	X	20	10

Informative Note: As of the 2015 revision of this specification the use of timecode in the VC-3 bitstream is deprecated, though still permitted.

Informative Note: VC-3 is not restricted to the limited set of video frame rates supported by SMPTE ST 12-1. If the TCP flag of Section 7.2.6 is not set, then any timecode may be specified as out-of-band information.

7.2.8 User Data Control

Figure 25 depicts the contents of the User Data Control area, which shall contain the User Data Label (UDL).

Byte Offset	**msb**							**Lsb**
0x05F	UDL_3	UDL_2	UDL_1	UDL_0	0	0	0	1

Figure 25 – User Data Control data area

UDL: User Data Label

A 4-bit field indicating the type of user data in the User Data Payload (Section 7.2.9).

The value of UDL shall be 0000b which indicates that no user data shall be contained in the User Data Payload. No other UDL values are defined by this document. They are reserved for future use.

See informative Annex B for a description of the intended use of the UDL field.

7.2.9 User Data Payload

The User Data Payload contains user data as indicated by the User Data Label of Section 7.2.8. When the value of UDL is 0000b, the entire contents of the User Data Payload shall be reserved and set to 0b.

7.2.10 Macroblock Scan Indices Control

As shown in Figure 26, the Macroblock Scan Indices Control data area shall contain a 16-bit representation of the value N_S.

Byte Offset	msb							Lsb
0x167	0	0	0	0	0	0	1	0
0x168	0	0	0	0	0	0	0	0
0x169	0	0	0	0	0	0	0	0
0x16A	$MSIPS_1$							
0x16B	$MSIPS_0$							
0x16C	Ns_1							
0x16D	Ns_0							
0x16E	0	0	0	0	0	0	0	0
0x16F	0	0	0	1	0	0	0	0

Figure 26 – Macroblock Scan Indices Control data area

N_S: Number of Scan Indices

The value of N_S shall specify how many macroblock scan indices are present in the Macroblock Scan Indices Payload area (Section 7.2.11).

$$N_S = (Ns_1 << 8) + Ns_0$$

MSIPS: Macroblock Scan Indices Payload Size

A two-byte field indicating the size of the Macroblock Scan Indices Payload area in bytes.

$MSIPS = (MSIPS_1 << 8) + MSIPS_0 - 4$

7.2.11 Macroblock Scan Indices Payload

The Macroblock Scan Indices Payload area shall contain a set of indices, one index for each macroblock scan line. The value of the index shall contain the starting byte offset of the Compressed Macroblock Data (Section 7.3.1.1) for the first macroblock in the macroblock scan line. The index offset value shall be relative to the start of the Compressed Payload at address 0x280.

The Macroblock Scan Indices Payload area shall be located at byte offset 0x170 upwards. The locations shall contain the four-byte macroblock scan indices. The indices shall be placed in this area in big-endian order, starting with the first macroblock scan line through the last macroblock scan line. The interlaced as well as the progressive video formats below 1080p do not require the full area since they have fewer macroblock scan lines than the 1080p video formats. In these cases, the remaining bytes of the data area shall be padded with 0 (even for VBR encoding).

An example of how macroblock scan indices are used is given in Section 7.5.

7.3 Compressed Payload

As shown in Figure 17, the Compressed Payload area shall consist of Compressed Data (Section 7.3.1) and Compressed Payload Padding (Section 7.3.2). This Compressed Payload area shall immediately follow the Header field shown in Figure 17.

7.3.1 Compressed Data

The Compressed Data shall consist of Compressed Macroblock Data (CMBD) and Macroblock Scan-line padding as shown in Figure 27. The starting address of each compressed macroblock scan line (MBSL) shall be a multiple of 4 bytes. The compressed macroblock data from the first macroblock shall be located right after the Macroblock Scan Indices Payload area followed by the compressed data of the succeeding macroblocks. At the end of a macroblock scan line (after N_W compressed macroblock data regions for video only, or 2 N_W compressed macroblock data sections for video + alpha) a section of Macroblock Scan-line Padding shall occur. This padding shall ensure that the start of the next compressed macroblock scan line occurs on a four-byte alignment boundary.

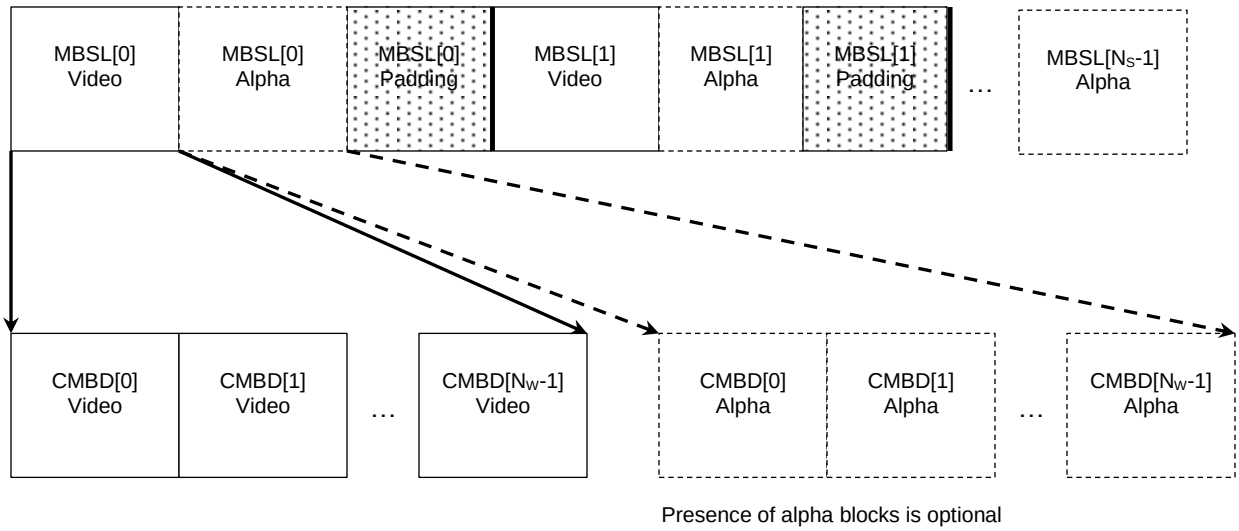

Figure 27 – Compressed Macroblock Data and padding format

A binary value of length *l* shall be mapped to the Compressed Data bitstream and byte-based memory as shown in Figure 28 (big-endian mapping).

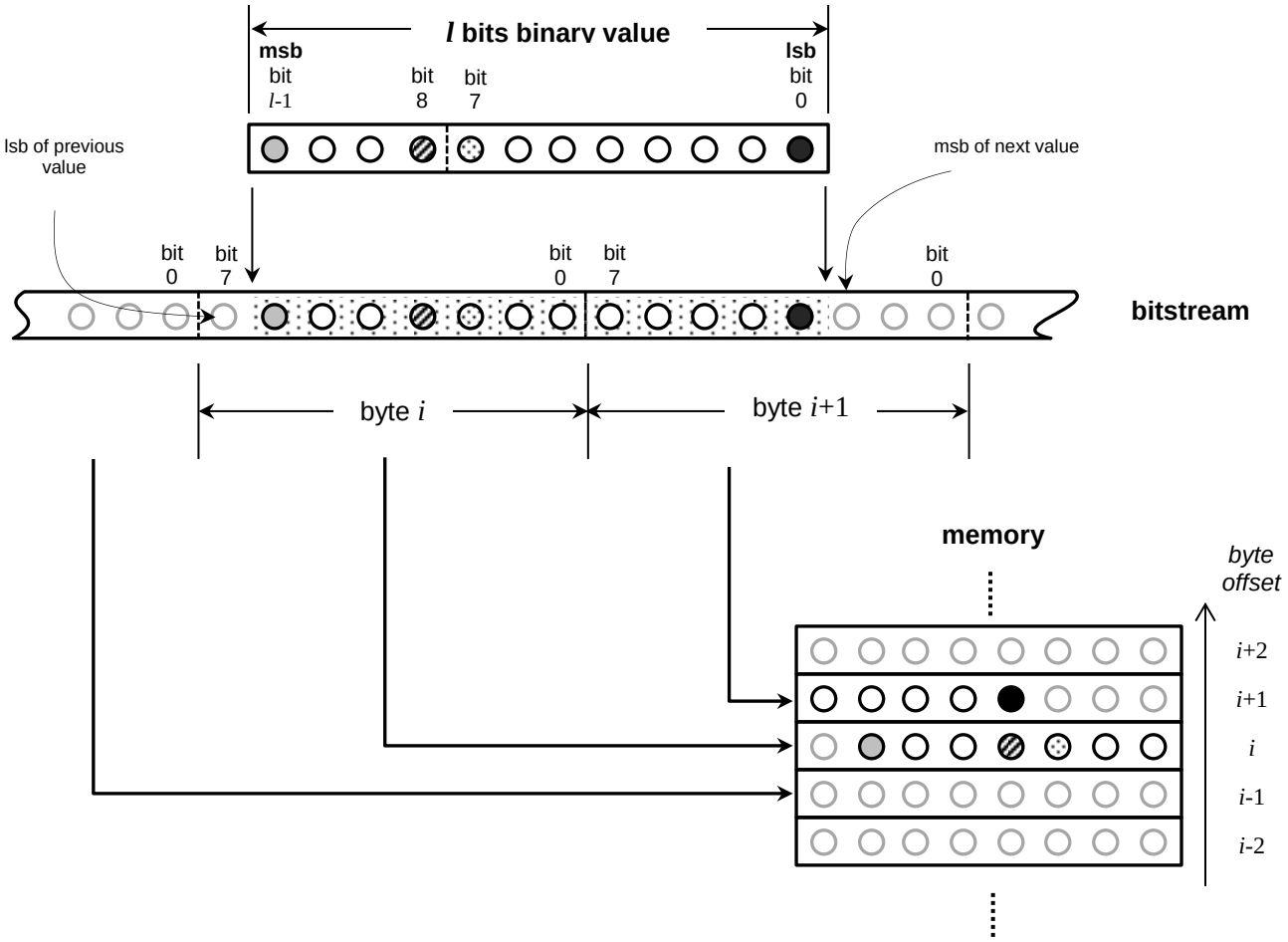

Figure 28 – Mapping of an *l*-bits value to bitstream and memory

7.3.1.1 DCT-Compressed Macroblock Data

The data (CMBD) of a DCT-compressed macroblock shall contain a 12-bit DCT-Macroblock Header, followed by the entropy codewords for each DCT block as diagrammed in Figure 29 Each DCT's entropy codeword bitstream shall be terminated by an End Of Block (EOB) codeword. At a minimum, a DCT's entropy codewords shall consist of a DC coefficient codeword and zero or more AC coefficient code words.

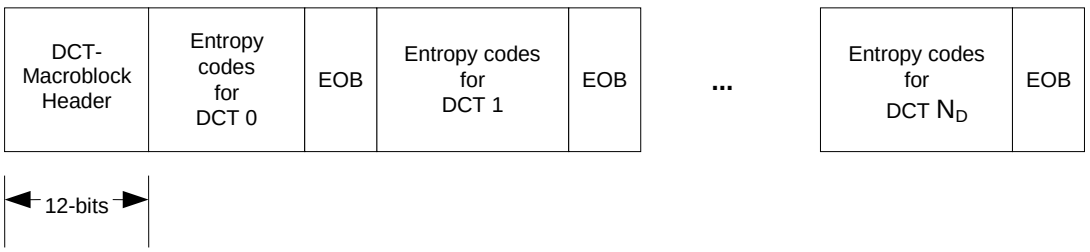

Figure 29 – DCT-Compressed Macroblock Data format

The contents of the macroblock header shall conform to one of three definitions, as shown in Figure 30, Figure 31 and Figure 32. Which definition shall be applied shall depend upon the Compression ID as indicated in Table 11.

Table 11 – Definition of Macroblock Header Format per Compression ID

Compression ID	Definition of Macroblock Header
Compression ID != 1256 and Compression ID != 1260 and Compression ID != 1270	As per Figure 30
Compression ID == 1256 or Compression ID == 1270	As per Figure 31
Compression ID == 1260	As per Figure 32

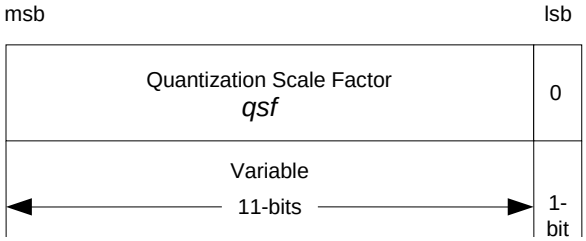

Figure 30 – Macroblock header format for bit-streams

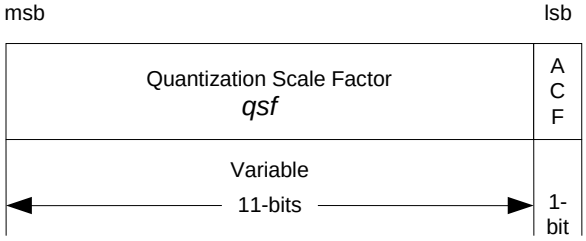

Figure 31 – Macroblock header format for bit-streams of CompressionIDs 1256, 1270

The interpretation of the ACF (Alternate Colorspace Flag) shown in Figure 31 shall be as follows:

ACF == 0b - Do not apply alternate colorspace transformation
ACF == 1b - Apply Recommendation ITU-R BT.709 YC_BC_R to RGB conversion for macroblock during decode

The value ACF==1b shall only be permitted for CLF==1 and CLV != 11b.

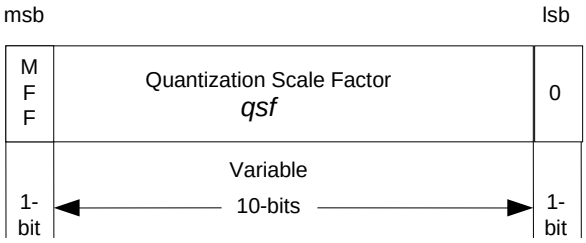

Figure 32 – Macroblock header format for bit-streams of Compression ID = 1260

The interpretation of the MFF flag shown in Figure 32 shall be as follows:

MFF == 0b - macroblock is in Frame Macroblock Mode as shown in Figure 13

MFF == 1b - macroblock is in Field Macroblock Mode as shown in Figure 14

7.3.1.2 RLE-Compressed Alpha Macroblock Data

The data (CMBD) of a RLE-compressed Alpha macroblock shall contain a 4-bit Alpha-Macroblock Header, followed by the variable number of codewords for the 256 samples for the Alpha macroblock as diagrammed in Figure 33. At a minimum 1 codeword shall follow the Alpha Macroblock header and 256 at a maximum.

RLE-compressed Alpha macroblocks shall only be permitted for CompressionID 1270.

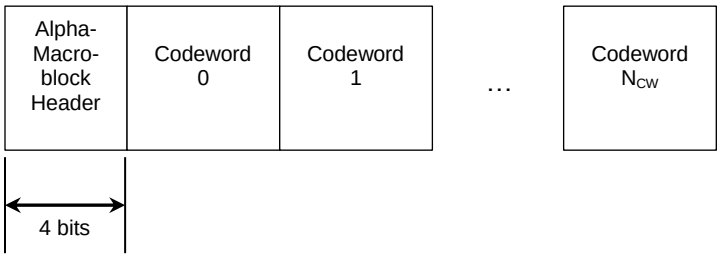

Figure 33 – RLE-Compressed Macroblock Data format

The format of the Alpha-Macroblock header shall conform to the definition in Figure 34.

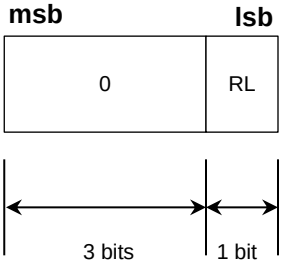

Figure 34 – Alpha Macroblock header codeword for CompressionID 1270

RL: RLE Indicator
The size of the codewords in bits following the Alpha Macroblock header shall not exceed

$$C_{lim} = D_{comp} \cdot 256$$

where D_{comp} is the bitdepth of the alpha component. If the RLE encoded block exceeds C_{lim}, then the whole block shall be stored as closely packed raw components instead.

RL = 0b: The following bitstream contains 256 raw components, each D_{comp} bits wide. The number of codewords in the alpha block N_{CW} shall equal 255.
RL = 1b: The following bitstream is RLE encoded.

Codewords for RL = 1b shall be encoded as follows

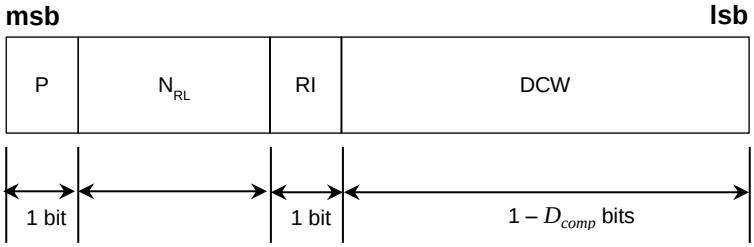

Figure 35 – Structure of RLE encoded codewords

P: Predictor indicator

Alpha values a_i are stored in differential form, where a difference d_i is stored instead a_i.

P = 0b: d_i is defined by $a_i = a_{i-1} + d_i$
P = 1b: d_i is defined by $a_i = 2a_{i-1} - a_{i-2} + d_i$

The differential reconstruction sequence shall start using $a_{-2} = a_{-1} = 2^{D_{comp}-1} - 1$.

N_{RL}: Run length of the fully decoded value a_i, encoded as an Elias Gamma (Exponential Golomb) universal code.

$N_{RL} = 2^N + R; \quad 0 \leq R \leq 2^N - 1$

N is the highest power of 2 contained in N_{RL} and R the remainder. These values shall be encoded as shown in Figure 36.

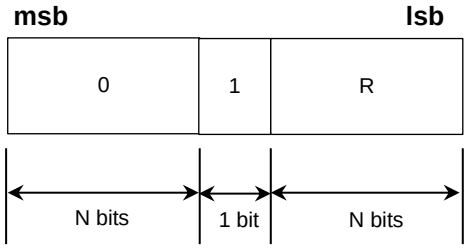

Figure 36 – Elias Gamma code for 2^N+R

Note: The run length can range between 1 and 256. The shortest Elias Gamma code is 1b (2^0 = 1), matching $N_{RL} \geq 1$. The lsb of R is located next to the 1 bit separator.

RI: Rice encoding indicator

RI = 0b: The following difference codeword DCW is a binary value of bitdepth D_{comp}.
RI = 1b: The following difference codeword DCW is using a Golomb-Rice universal code.

DCW: Difference codeword

The difference d_i is first mapped to the unsigned DCW according to

$$\mathrm{DCW} = \begin{cases} 2d_i & ; d_i \geq 0 \\ -2d_i - 1 & ; d_i < 0 \end{cases}$$

For Golomb-Rice encoding (RI=1b) the DCW is then expressed as

$$\mathrm{DCW} = Q \cdot 2^r + R \quad ; \ 0 \leq R \leq 2^r - 1, \ r = (D_{comp}/2) - 2$$

where r is the Rice order, defined by the component bitdepth, Q the truncated quotient obtained by $Q = \lfloor \mathrm{DCW}/2^r \rfloor$ and R the remainder. These values shall be encoded as shown in Figure 37.

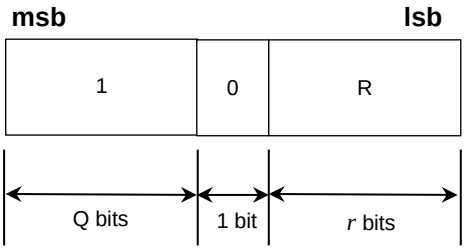

Figure 37 – Rice encoding of the codeword value DCW

7.3.1.3 Macroblock Scan-Line Padding

Macroblock scan-line padding shall occur at the end of macroblock scan-line row. Since the starting address of each compressed macroblock scan line shall be a multiple of four bytes, the length of padding can vary between 0 and 31 bits. In the case where a compressed macroblock scan line ends on a multiple of four bytes no padding shall be required. The padding value shall be 0b, for each padding bit.

Note: The amount of padding is variable, depending on the total length of the compressed macroblock data sections.

7.3.2 Compressed Payload Padding

Compressed Payload Padding shall fill the Compressed Payload Area that is not consumed by the Compressed Data for CBR encoding. The compressed payload padding shall be filled with byte values 0.

For VBR encoding this block shall be absent from the Compressed Payload Area.

Note: The number of bytes of Compressed Payload Padding is variable as the number of compressed bytes depends on video content and visual quality requirements.

7.4 EOF Signature

The contents of the four-byte EOF signature region shall depend on the value of the CRC flag (CRCF), as defined in Section 7.2.2.

If CRCF = 1, then the region shall contain the value shown in Figure 38.

The CRC algorithm for the EOF signature shall be the 32-bit CRC defined by the generator polynomial (0x04C11DB7 [msbit-first, (normal) CRC-generator notation]), as defined for Ethernet use by IEEE 802.3:

$$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

with an initial seed value of 0, storing the result in big-endian form as:

Byte Offset	msb lsb
0x0	CRC_3
0x1	CRC_2
0x2	CRC_1
0x3	CRC_0

Figure 38 – EOF Signature contents for CRCF = 1

If CRCF = 0, then the EOF signature region shall contain the value shown in Figure 39.

Byte Offset	msb lsb
0x0	CRC_3 = 0x60
0x1	CRC_2 = 0x0D
0x2	CRC_1 = 0xC0
0x3	CRC_0 = 0xDE

Figure 39 – EOF Signature contents for CRCF = 0

7.4.1 **Verification of Data Integrity** (Informative)

To verify the integrity of the data in the coding unit, the CRC computation is initialized with the value of 0. The data contained in the coding unit, including the four (CRC) bytes in the EOF signature area, is presented to the CRC computation as a byte stream, based on the sequential byte storage order. Upon completion of the last CRC calculation, a CRC output value other than 0 indicates data corruption.

7.5 Illustrative Example (Informative)

The following example provides the contents of the macroblock scan indices and the compressed macroblock starting addresses for a hypothetical coding unit. It also computes the size of the Compressed Payload Padding for CBR encoding. The coding unit represents a field of interlaced 1920x1080 encoding using Compression ID 1242. The columns in Table 12 list pertinent information.

In this example, the first compressed macroblock scan line requires 7985 bits to encode. A total of 15 padding bits are present at the end of the compressed macroblock scan line, so that the end of the compressed data is a multiple of four bytes. The length of the compressed macroblock and padding is 1000 bytes. The first macroblock scan-line index payload area is at location 0x170. It contains the value 0 to indicate that the first macroblock scan area starts at byte location 640 (0x0280).

The second macroblock scan line index payload area is located at byte location 0x174. It contains the value 1000 which indicates that the start of the second macroblock scan line is at byte location 1640 (0x0668).

Since this example compression involves interlaced video, Table 3 indicates that N_S = 34. Thus, the last value specified in the macroblock scan line index payload is location 0x1F4. The unused locations 0x1F8 through 0x27F are filled with padding.

The total compressed data consumes 300400 + 1600 = 302000 bytes. As listed in Table C.1, the compressed payload size is 302460 bytes. The remainder of the compressed payload consists of compressed payload padding, which means that 302460 – 302000 = 460 bytes of compressed payload padding are present in the coding unit.

Table 12 – Contents of Macroblock scan-line indices for the illustrative example

Macro block scan line	Length of compressed macroblock scan line (bits)	Length of macroblock scan-line padding (bits)	Length of compressed macroblock scan-line and padding (bytes)	Macroblock scan line index location	Contents of Macroblock scan-line index	Starting address of compressed macroblock scan line
0	7985	15	1000	0x170	0x0000 0000 (0)	0x0000 0280 (640)
1	9569	31	1200	0x174	0x0000 03E8 (1000)	0x0000 0668 (1640)
2	…	…	…	0x178	0x0000 0898 (2200)	0x0000 0B18 (2840)
…	…	…	…	…	…	…
32	6400	0	800	0x1F0	0x0004 9250 (299600)	0x0004 9368 (299880)
33	12789	11	1600	0x1F4	0x0004 9570 (300400)	0x0004 97F0 (301040)

8 Decoding

A decoder shall inspect the Header (Section 7.2) prior to decoding and proceed if and only if the HVN (section 7.2.1) and Compression ID (Section 7.2.4) are explicitly known to the decoder implementation.

The decoding process shall convert the compressed macroblock data into video raster data. The process is summarized in the pseudo code listed in Figure 40. The range of the macroblock index per type shall be l = 0, 1, …, $N_T - 1$. The pseudo code listing contains 3 functions: `decodeMacroblock()`, `decodeAlphaMacroblock()` and `skipMacroblockPadding()`.

The `decodeMacroblock()` function is described in Section 8.1, the `decodeAlphaMacroblock()` function will be described in Section 8.3. The `skipMacroblockPadding()` function removes macroblock scan padding bits in CBR mode. This shall occur for l=N_W-1, 2N_W-1, …, (N_S-1)N_W-1.

```
l=0; // Macroblock index
for ( s=0 ; s<Ns ; s++ )
{
  for ( w=0 ; w<NW ; w++, l++ )
    decodeMacroblock(MBl,l);

  // ALP = Alpha flag
  if (ALP == 1)
  {
    for ( w=0 ; w<NW ; w++, l++ )
      decodeAlphaMacroblock(AMBl,l);
  }
  if (VBR == 0b)
    skipMacroblockPadding();
}
```

Figure 40 – Pseudocode for the decoding process

8.1 Macroblock Decoding

8.1.1 Video

The Video macroblock decoding process shall convert the compressed macroblock data for MB_l to its corresponding video raster data. Figure 41 provides the pseudo code representation of the process.

```
decodeMacroblock(MB_l, l)
{
  // k = DCT index, n = number of blocks within a macroblock
  for ( k=0 ; k<n ; k++)
  {
    x_k^l = decodeDCTBlock(v_k^l, k, l);
    forall ( [0,0] <= [i,j] <= [7,7] )
    {
      // CID = Compression ID
      if ( (CID == 1256) || (CID == 1270) )
      {
        // RGB/4:4:4
        switch (k)
        {
          case 0,1,6,7:     Ch1_k^l[i,j]= x_k^l[i,j]; break;
          case 2,3,8,9:     Ch2_k^l[i,j]= x_k^l[i,j]; break;
          case 4,5,10,11:   Ch3_k^l[i,j]= x_k^l[i,j]; break;
        }

        if (ACF == 1)
          ConvertMacroblockColor(MB_l, l);
      }
      else
      {
        // 4:2:2 and 4:2:0
        switch (k)
        {
          case 0,1,4,5:  Y_k^l[i,j]  = x_k^l[i,j]; break;
          case 2,6:      Cb_k^l[i,j] = x_k^l[i,j]; break;
          case 3,7:      Cr_k^l[i,j] = x_k^l[i,j]; break;
        }
      }
    }
  }
}
```

Figure 41 – Macroblock decoding pseudo code

The function `ConvertMacroblockColor(`MB_l`, l)` shall convert the pixel data contained in Ch1, Ch2, and Ch3 from YC_BC_R data to RGB data for macroblock MB_l according to the signal transformation specified in Recommendation ITU-R BT 709, Part 2, Section 3 (Signal format), item 3.5 for the quantized signals.

The DCT decoding function `decodeDCTBlock(`v_k^l`, k, l)` converts the variable-length bitstream v_k^l into a set of 2-D, decompressed video samples $x_l^k(i,j)$; $i,j = 0,1,\ldots,7$. The decoding process is depicted in Figure 42. It should consist of the following steps:

1 Entropy decoding: Entropy decoding (Section 8.2.1) converts v_k^l into a set of quantized DCT coefficients $\hat{X}_k^l(r)$. The coefficients shall be ordered in bit-stream order, denoted by bitstream coefficient index, r.

2 Inverse Zig-zag: The inverse zig-zag process (Section 8.2.6) reorders the quantized DCT coefficients $\hat{X}_k^l(r)$ into a 2-D array $\hat{X}_k^l(u,v)$

3 Inverse quantization: During the inverse quantization process (Section 8.2.7), the quantization scaling factor, qsf, which shall be encoded in the macroblock header (Section 8.2.1), and the quantization weights, $W(u,v)$, shall be applied to compute a set of DCT coefficients $X_k^l(u,v)$

4 Inverse Discrete Cosine Transform: The Inverse DCT process (Section 8.2.8.1) transforms the 2-D DCT coefficients $X_k^l(u,v)$ to a set of 2-D, decompressed video samples $x_k^l(i,j)$.

5 Adjust the level of the IDCT output as described in Section 8.2.8.3.

The decoding process for a single DCT block is detailed in the following sub-sections. The subscript k (DCT index) and the superscript l (macroblock index) are dropped from the notation for readability.

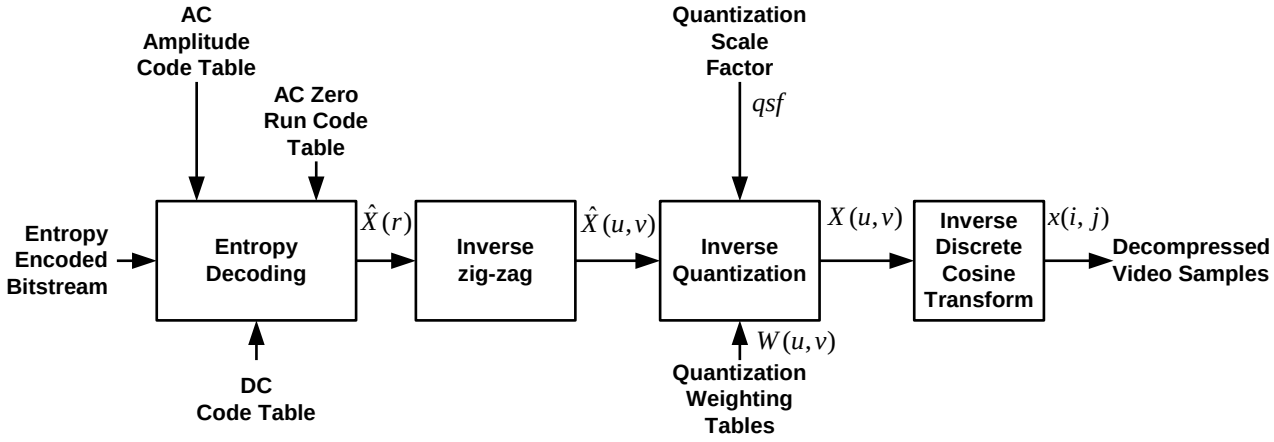

Figure 42 – Decoding block diagram for a single DCT block

8.1.2 Alpha

The Alpha macroblock decoding process shall convert the compressed Alpha macroblock data for MB_l to its corresponding video raster data. Figure 43 provides the pseudo code representation of the process.

```
decodeAlphaMacroblock(AMB_l, l)
{
  if (LLA == 1)
  {
      A^l = decodeRLEBlock(v^l, l);
  }
  else
  {
    for ( k=0 ; k<4 ; k++)
    {
      A_k^l = decodeDCTBlock(v_k^l, k, l);
    }
  }
}
```

Figure 43 – Alpha-Macroblock decoding pseudo code

The function `decodeDCTBlock(`v_k^l`, k, l)` is the same DCT decode function used to decode the Y channel (Section 8.1.1) for the CompressionID.

The lossless decoding function `decodeRLEBlock(`v^l`, l)` used for CompressionID 1270 if the lossless alpha compression flag *LLA* is set, will be described in Section 8.3.

8.2 DCT block decoding

The DCT block decoding is broken down in multiple stages below.

8.2.1 Entropy Decoding

As shown in Figure 44, the elements of a DCT block's encoded bitstream consist of a grouping of entropy codewords. The codeword types fall into three categories: DC DCT coefficient codewords (v_{DC}), AC DCT coefficient codewords ($\mathbf{v}_{AC}$), and EOB codewords (v_{EOB}). This is represented by the notation:

$$\mathbf{v} = \{v_{DC}, \mathbf{v}_{AC}, v_{EOB}\}$$

The following sub-sub sections describe the entropy decoding process.

Figure 44 – A DCT block's entropy bitstream

8.2.2 Macroblock Header and Quantization Scale Factor

The macroblock header shall be a 12-bit data field that contains a quantization scale factor *qsf*.

The format for *qsf* shall follow one of the Figure 30, Figure 31 or Figure 32, depending on the Compression ID.

8.2.3 VLC Codeword Tables

For each compression ID, three sets of variable-length codeword (VLC) tables shall be defined: The amplitude codeword table V^A, the zero run-length table V^R, and the DC VLC table V^{DC}. The entries for these tables are listed in Annex E: VLC Tables, according to compression ID.

The amplitude codeword tables are found in Table E.1, Table E.4, Table E.7, Table E.10, Table E.13 and Table E.16.

The zero run-length tables are found in Table E.2, Table E.5, Table E.8, Table E.11, Table E.14 and Table E.17.

The DC coefficient decoding process utilizes the DC VLC Table E.3, Table E.6, Table E.12, Table E.15 and Table E.18.

8.2.4 DC Coefficient Decoding

Computing the DC coefficient of a DCT block $\hat{X}_k(0)$ shall follow a multi-step process. First, the DC coefficient codeword v_{DC} shall be decoded. The results of the decoding enable the computation of a prediction correction value ε. This prediction correction value shall be added to a DC coefficient prediction value P_{ct} to compute $\hat{X}_k(0)$. This process is further described below.

v_{DC} shall consist of two elements as shown in Figure 45. The first element shall be a variable-length codeword indicating the number of bits η required to decode the second. The second element, designated ρ, shall be an intermediate binary value used to compute ε. The mapping from a variable-length codeword to a value of η shall be as defined in the DC VLC Tables listed in Annex E VLC Tables (Normative).

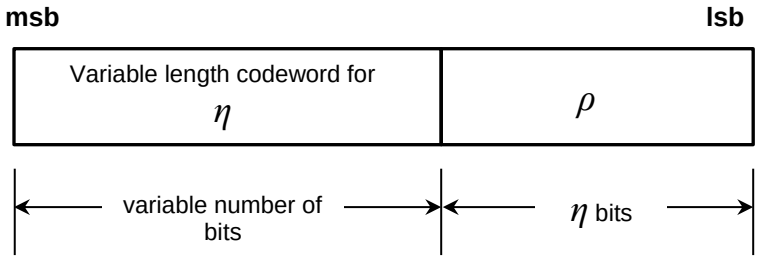

Figure 45 – Format for the DC codeword v_{DC}

$$\text{Range} = \begin{cases} \pm(2^{\eta}-1), \pm(2^{\eta}-2), \ldots, 2^{(\eta-1)} & ;\eta \geq 2 \\ \pm 1 & ;\eta = 1 \\ 0 & ;\eta = 0 \end{cases}$$

For 8-bit video sampling, the maximum value of η = 11 and for 10-/12-bit video sampling, the maximum value of η= 13.

Summarizing:

- ε shall be the prediction correction value.
- ρ shall be an intermediate binary value used to compute ε. It shall be an unsigned binary value with a length of η.
- η shall be the length of ρ in bits.
- The variable-length codewords for η shall be as listed in the DC VLC table given in Annex E: VLC Tables.
- ε shall be computed from ρ according to the pseudo code given in Figure 46.

Table 13 – Relationship between the range of the correction prediction value and the length of ρ

Range of prediction correction value ε	η: Length of ρ in bits
0	0
+/- 1	1
+/- 3, +/- 2	2
+/- 7, …, +/- 4	3
+/- 15, …, +/- 8	4
+/- 31, …, +/- 16	5
+/- 63, …, +/- 32	6
+/- 127, …, +/- 64	7
+/- 255, …, +/- 128	8
+/- 511, …, +/- 256	9
+/- 1023, …, +/- 512	10
+/- 2047, …, +/- 1024	11
+/- 4095, …, +/- 2048	12
+/- 8191, …, +/- 4096	13

```
if (η = 0)
  ε = 0;
else
{
  if (ρ >= 2^(η-1))
    ε = ρ;
  else
    ε = ρ+1-2^η;
}
```

Figure 46 – Pseudo code for computation of the prediction correction value ε

A separate prediction value P_{ct} shall be maintained for each component type (ct = Y, C_B, C_R, R, G, B or A). P_{ct} shall be set to 0 at the beginning of a macroblock scan line for each component type. The DC coefficient for DCT block k (with component type ct) shall be computed by:

$$\hat{X}_k(0) = P_{ct} + \varepsilon_k$$

The prediction value shall be updated according to:

$$P_{ct} \leftarrow \hat{X}_k(0)$$

except at the beginning of a macroblock scan line.

8.2.5 AC Coefficient Entropy Decoding

The process of AC coefficient entropy decoding shall extract the quantized AC DCT coefficients from the entropy encoded bitstream. The entropy codewords shall represent the sign and amplitude of the non-zero values quantized AC coefficients and the run lengths of zero-valued AC coefficients. A description of the format of AC coefficient entropy codewords is given in Annex A.

The AC coefficient entropy codewords shall be a set defined by:

$$\mathbf{v}_{AC} = \begin{cases} \emptyset & \text{if no AC coefficients are encoded} \\ \{v_1, v_2, \ldots, v_N\} & \text{otherwise} \end{cases}$$

where $\emptyset$ is the empty set and *N* is the number of non-zero AC coefficients in the quantized DCT.

8.2.5.1 AC Coefficient Codeword Tables

The amplitude codeword table V^A shall provide the following information for each unique codeword. The amplitude codeword tables shall be as listed in Annex E: VLC Tables.

- The length of the codeword in bits shall correspond to the “Length” column.
- The amplitude of the quantized DCT coefficient shall correspond to the “Amp” column.
- A flag F_{run} shall indicate if there are one or more zero-valued coefficients preceding the coefficient. This flag shall correspond to the “Run Flag F_{run}” column.
- A flag F_{index} shall indicate that the amplitude of the coefficient is to be adjusted by an offset. This flag shall correspond to the “Index Flag F_{index}” column.

8.2.5.2 AC Coefficient Decoding Pseudo Code

The AC coefficients shall be decoded in a manner which gives identical results to the process defined in Section 8.2.5.3.

8.2.5.3 AC Coefficient Decoding Pseudo Code Process (Informative)

The pseudo code shown in Figure 47 describes an algorithm for decoding the AC coefficients of one DCT block. This algorithm provides the quantized coefficients $\hat{X}(r)\ for\ r = 1,2,\ldots,63$ by decoding the

codewords in $\mathbf{v}_{AC}$. The pseudo code makes use of three functions: `decode_VLC_VA()`, `decode_P()`, and `decode_VLC_VR()`.

The function `decode_VLC_VA()` operates on the amplitude codeword table V^A to provide:

- A – the value from which the coefficient's amplitude is computed.
- S – the sign of the coefficient. $S = \begin{cases} -1 & \text{for } v_i = 1 \\ 1 & \text{for } v_i = 0 \end{cases}$
- F_{run} – a flag that, if set to 1, indicates that one or more zero value coefficients precede the current coefficient.
- F_{index} – a flag that, if set to 1, indicates that the coefficient amplitude shall be adjusted by an offset, dependent on the amplitude index P. The amplitude index is encoded later in the bitstream.
- F_{EOB} – a flag that, if set to 1, indicates that the end of bitstream (EOB) has occurred.

The function `decode_P()` is applied when F_{index} is true. It returns P, which is decoded from the bit-stream of v_i.

The function `decode_VLC_VR()` operates on the zero run-length table V^R to provide the number of zero-valued coefficients preceding the current coefficient. It is when F_{run} is 1.

```
for ( r=1 ; r<64 ; r++)
  X̂(r) = 0;

// r = DCT coefficient index, i = Codeword index
for ( r=1,i=1 ; r<64 ; r++,i++ )
{
  (A, S, F_run, F_index, F_EOB) = decode_vlc_VA(v_i);
  if (F_EOB)
    break;

  if (F_index)
  {
    P = decode_P(v_i);
    A = A + P*64;
  }

  if (F_run)
  {
    Run = decode_vlc_VR(v_i);
    r   = r+Run;
  }

  X̂(r) = S*A;
}
```

Figure 47 – Pseudo code for AC coefficient entropy decoding of a single DCT block

8.2.6 Inverse Zig-zag

The conversion from bitstream order index r to 2-D DCT array indices (u,v) shall be as shown in Figure 48. This conversion remaps the quantized coefficients $\hat{X}(r)$ to $\hat{X}(u,v)$.

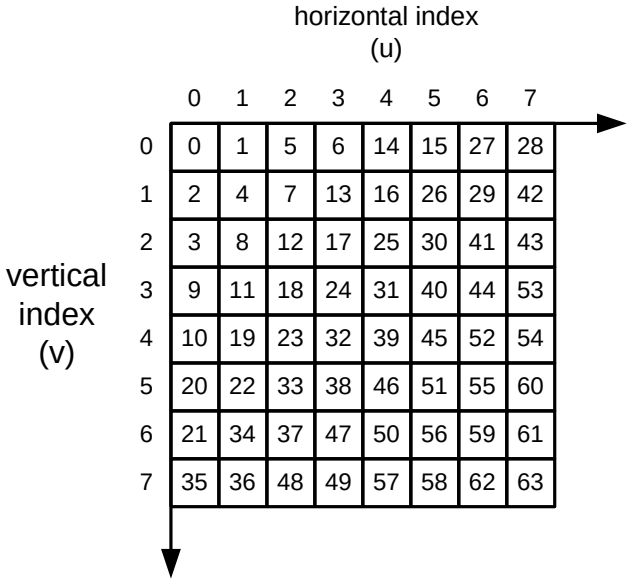

Figure 48 – Zig-zag ordering $\mathrm{r} \rightarrow (\mathrm{u,v})$

8.2.7 Inverse Quantization

Computation of the inverse quantization of a set of quantized DCT values $\hat{X}(u,v)$ shall utilize the quantization scale factor (qsf) present in the macroblock header and a set of quantization weights $W(u,v)$. The weights for each Compression ID shall be as listed in Annex D: Quantization Weights. In the cases of Compression ID 1256 and Compression ID 1270 all three channels shall use the same quantization weights. For other compression IDs the luma and color-difference components shall each have their own set of quantization weights.

Inverse quantization shall not be applied to the DC coefficient $\hat{X}(0,0)$, since it is not quantized during the encoding process. In other words:

$$X(0,0) = \hat{X}(0,0).$$

The inverse quantization process for AC coefficients shall be as shown in (8.1)

if $\hat{X}(u,v) = 0$
$\quad X(u,v) = 0$
else
{

if $W(u,v) \neq p$

$$X(u,v) = \text{sgn}(\hat{X}(u,v)) \left\lfloor \frac{\left|\hat{X}(u,v)\right| \cdot W(u,v) \cdot qsf + \left\lfloor \frac{W(u,v) \cdot qsf}{2} \right\rfloor + \frac{p}{2}}{p} \right\rfloor$$

else **(8.1)**

$$X(u,v) = \text{sgn}(\hat{X}(u,v)) \left\lfloor \frac{\left|\hat{X}(u,v)\right| \cdot W(u,v) \cdot qsf + \left\lfloor \frac{W(u,v) \cdot qsf}{2} \right\rfloor}{p} \right\rfloor$$

}

with the inverse quantization parameter p

$$p = \begin{cases} 8, & \text{for Compression ID} = 1235,1241,1250 \\ 32, & \text{for all other.} \end{cases}$$

Note : The quantization function, as used during *encoding*, is shown in equation (8.2)

$$\hat{X}(u,v) = \text{sgn}\big(X(u,v)\big) \left\lfloor \frac{|X(u,v)|}{\frac{qsf}{p} \cdot W(u,v)} \right\rfloor = \text{sgn}\big(X(u,v)\big) \left\lfloor \frac{p \cdot |X(u,v)|}{qsf \cdot W(u,v)} \right\rfloor \quad \textbf{(8.2)}$$

The encoder is responsible to select appropriate *qsf* values for each macro block to meet the normative frame sizes for each compression ID.

8.2.8 Discrete Cosine Transform

8.2.8.1 8x8 Inverse DCT

The IDCT shall produce a set of 8x8 video pixels from the DCT coefficients $X(u,v)$. The 8x8 Inverse DCT of DCT coefficients $X(u,v)$, $u,v = 0,\ldots,7$ shall be defined according to:

$$x(i,j) = (1/4)\sum_{u=0}^{7}\sum_{v=0}^{7} C(u)C(v)X(u,v)\cos\left(\frac{(2i+1)\pi u}{16}\right)\cos\left(\frac{(2j+1)\pi v}{16}\right) \quad \textbf{(8.3)}$$

8.2.8.2 8x8 Forward DCT (Informative)

For reference the Forward Discrete CosineTransform (FDCT) is given below. The 8x8 Forward DCT of pixel data $x(i,j), i, j = 0,...,7$ produces an 8x8 array of DCT coefficients:

$$X(u,v) = (1/4)C(u)C(v)\sum_{i=0}^{7}\sum_{j=0}^{7} x(i,j)\cos\left(\frac{(2i+1)\pi u}{16}\right)\cos\left(\frac{(2j+1)\pi v}{16}\right)$$

where **(8.4)**

$C(0) = 1/\sqrt{2}$ and $C(u) = C(v) = 1$, for $u,v = 1,...7$

8.2.8.3 Video Sample Level Adjustment

As shown in Table 14, the dynamic range of the data increases to 11/13 bits respectively when pixel data passes through the DCT, and decreases accordingly when going through the IDCT.

Table 14 – Dynamic ranges of FDCT and IDCT for 8-, 10- and 12-bit video sampling depth

	FDCT Range		IDCT Range	
	Input	Output	Input	Output
8-bit sampling depth	-128 to 127 (8 bits)	-1024 to 1023 (11 bits)	-1024 to 1023 (11 bits)	-128 to 127 (8 bits)
10-bit sampling depth	-512 to 511 (10 bits)	-4096 to 4095 (13 bits)	-4096 to 4095, (13 bits)	-512 to 511 (10 bits)
12-bit sampling depth	-2048 to 2047 (12 bits)	-4096 to 4095 (13 bits)	-4096 to 4095, (13 bits)	-2048 to 2047 (12 bits)

The last stage of DCT decoding should be to adjust the levels of the video data. The adjustment value d shall be $d = 2^{b-1}$, where b is the sample bit depth value determined from SBD field in the Image Geometry header data area described in Section 7.2.3. The adjustment process is described by the pseudo code given in Figure 49.

```
for ( i=0 ; i<8 ; i++)
  for ( j=0 ; j<8 ; j++)
    x[i,j] = x[i,j]+ d;
```

Figure 49 – Pseudo code for level adjustment of a DCT block.

8.3 RLE Block Decoding

The function `decodeRLEBlock(`v^l`,l)` converts the variable-length bitstream v^l into a set of 2-D, decompressed video samples $A^l(i,j)$; $i,j = 0,1,\ldots,7$. The block may either be encoded completely as tightly packed binary values of component bit depth D_{comp} or as differential RLE.

```
decodeRLEMacroblock(v^l,l)
{
  mask = 2^Dcomp - 1;

  if ( (getbits(v^l,4) & 1) == 1) // Block encoded as binary
  {
    for ( count=0 ; count <256 ; count++ )
    {
      i,j    = inverseAlphaZigZag(count);
      A^l[i,j] = getbits(v^l,Dcomp);
    }
  }
  else
  {
    count = 0;
    D_1   = D_2 = mask >> 1;     // Block encoded as RLE
```

```
        while ( count < 256 )
        {
          p    = getbits(v^l,1);
          N_RL  = getEliasGammaDecode(v^l);

          if (getbits(v^l,1) == 1)
            DCW = getRiceDecode(v^l);     // Rice encoding
          else
            DCW = getbits(v^l,D_comp);    // Binary encoding

          if (DCW & 1)
            d = mask - (DCW >> 1);
          else
            d = DCW >> 1;

          for ( k=0 ; k<N_RL ; k++, count++ )
          {
             i,j = inverseAlphaZigZag(count);

            // Predictor-based reconstruction
            if (p == 1)
              A^l[i,j] = (2*D_1 - D_2 + d) & mask;
            else
              A^l[i,j] = (D_1 + d) & mask;

            D_2 = D_1; D_1 = A^l[i,j];
          }
        }
      }
    }
```

Figure 50 – Differential RLE decoding pseudo code

The pseudo code above presumes unsigned integer treatment throughout, with the operation `x & mask` representing wrap-around for unsigned integers of bitdepth D_{comp} executed on unsigned (working) integers of a higher bitdepth.

8.4 Best Practices (Informative)

Following the loss or corruption of part of a bitstream, it is suggested that a conformant decoder resume the decoding process as soon as possible. It is practical to resynchronize the decoder on a subsequent frame boundary.

In order to preserve color fidelity, video processing systems which interface to a VC-3 decoder could take special precautions when down-converting from a higher sample bit depth to a lower one (by applying suitable rounding or dithering algorithms) or when converting between color volumes (Recommendation ITU-R BT.2020 to Recommendation ITU-R BT.709 or vice versa).

Annex A AC Coefficient Entropy Codeword Format (Normative)

This annex pertains to the format of quantized AC coefficient entropy codewords. For the remainder of this annex, the term coefficient means quantized AC coefficient.

A.1 Entropy Codeword Symbol Set Definitions:

VC-3 shall utilize six types of symbol sets to represent coefficients: four for non-zero coefficient amplitudes, one for run lengths of zero-valued coefficients, and one for the End of Block condition. A coefficient's codeword format shall depend upon which symbol set is used to represent the coefficient. In order to describe the symbol sets, the following definitions are given.

- Coefficients residing in the "base range" of amplitude shall have amplitudes between 1 and 64 inclusive.
- Coefficients residing in the "index range" of amplitude shall have amplitudes between 65 and 4096 inclusive.
- The index range shall be partitioned into a number of segments, each of length 64.
- The index value P shall indicate from which segment a coefficient in the index range originates. The relationship between P and the coefficient amplitude A shall be:

  $P = \left((A-1) \gg 6\right) \qquad \text{for } 65 \le A \le 4096$

- If the video sampling depth is 10- or 12-bits, then P shall be a 6-bit binary value indicating the amplitude offset value. P shall be a 4-bits binary value, when the video sampling is 8-bits.

The six symbol sets shall be defined as follows:

1. $A^{nrb} = \{A_1^{nrb}, A_2^{nrb}, ..., A_{64}^{nrb}\}$: Non-zero coefficient amplitudes in the base range, with *no* preceding run of zero-valued coefficients. The superscript nrb represents no-run, base amplitude. The amplitudes shall vary from $A_1^{nrb} = 1$ to $A_{64}^{nrb} = 64$.

2. $A^{wrb} = \{A_1^{wrb}, A_2^{wrb}, ..., A_{64}^{wrb}\}$: Non-zero coefficient amplitudes in the base range, *with* preceding run of zero-valued coefficients. The superscript wrb represents with-run, base amplitude. The amplitudes shall vary from $A_1^{wrb} = 1$ to $A_{64}^{wrb} = 64$.

3. $A^{nri} = \{A_1^{nri}, A_2^{nri}, ..., A_{64}^{nri}\}$: Non-zero coefficient amplitudes in the index range, with *no* preceding run of zero-valued coefficients. The superscript nri represents no-run, index amplitude. The amplitudes shall vary from 65 to 4096.

4. $A^{wri} = \{A_1^{wri}, A_2^{wri}, ..., A_{64}^{wri}\}$: Non-zero coefficient amplitudes in the index range, *with* preceding run of zero-valued coefficients. The superscript wri represents with-run, index amplitude. The amplitudes shall vary from 65 to 4096.

5. $R = \{R_1, R_2, ..., R_{\max}\}$: a run of 1 or more zero-valued coefficients. $R_1 = 1$ and $R_{\max} = 62$.

6. $E = \{EOB\}$: the end of block symbol.

Figure A.1 shows the mapping between coefficients and the sets A^{nrb}, A^{nri}, A^{wrb}, A^{wri} and R.

The symbols in the five sets of $A^{nrb}, A^{nri}, A^{wrb}, A^{wri}, E$ shall be grouped together represented by a set of variable-length code words $V^A = \{V^{nrb}, V^{nri}, V^{wrb}, V^{wri}, V^E\}$. There shall be $4*64+1=129$ code words in V^A. The remaining symbol set, R, shall be represented by its own set variable-length, zero-run code words V^R. There shall be 62 codewords in V^R.

A.2 AC Coefficient Entropy Codeword Format

If a coefficient is an element of the set A^{nrb} (that is., the dotted region of Figure A.1), it shall be represented by a single code word taken from V^{nrb} followed by a 1-bit sign flag indicating the coefficient's sign value. The entropy encoding format for this case is shown at the top of Figure A.2.

If a coefficient resides in the set A^{wrb}, it shall be represented by a codeword taken from V^{wrb}, followed by a 1-bit sign flag, followed by a codeword taken from V^R. This case is depicted in the drawing that is second from the top of Figure A.2.

If a coefficient has no preceding run of zeros and lies in the index range, it shall be an element of A^{nri}. In this case, the coefficient shall be represented by a codeword taken from V^{nri}, followed by a 1-bit sign flag, followed by the index value P as shown in the middle drawing of Figure A.2.

A coefficient that is a member of the set A^{wri} shall be represented by a codeword taken from V^{wri}, followed by a 1-bit sign flag, followed by the index value P, followed by a codeword taken from V^R to represent the number of preceding zero runs. This format is shown in the drawing that is third from the top of Figure A.2.

Finally, the end of block case shall be represented as shown at the bottom of Figure A.2.

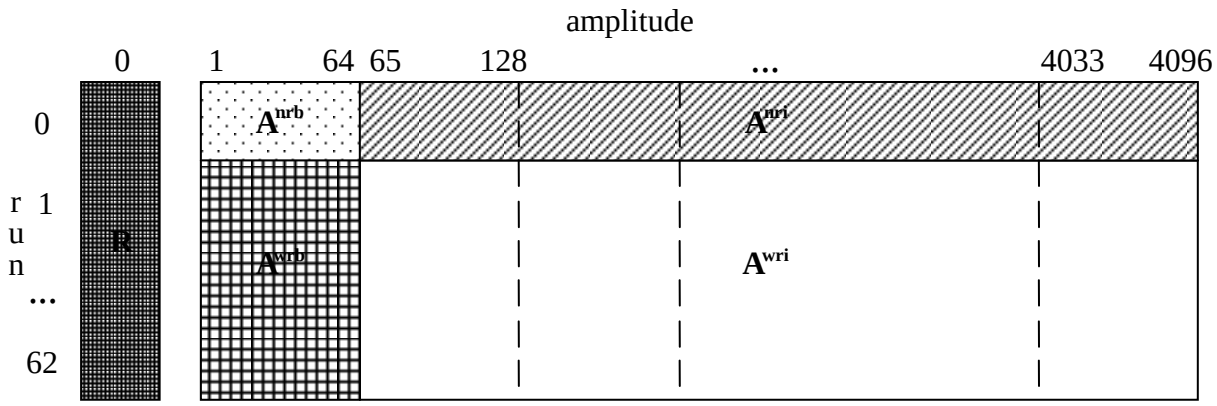

Figure A.1 – Mapping of Zero Run/Amplitude Combinations to Symbol Sets

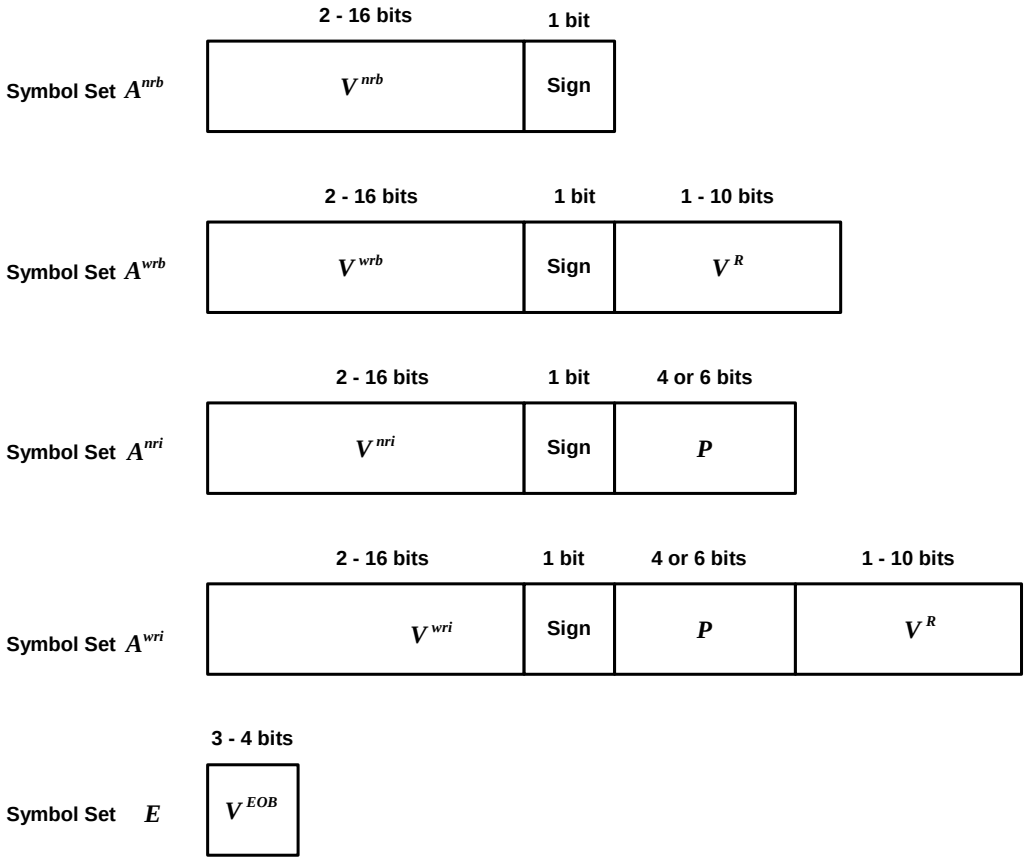

Figure A.2 – Symbol set formats

A.3 Description of the Entropy Codeword Format (Informative)

The categorization of non-zero coefficient amplitudes into one of the four possible sets depends on the amplitude value (A) and the absence or presence of preceding zero-valued coefficients.

The range of A is partitioned into two sub-ranges: a base range for amplitudes between 1 and 64 and an index range for amplitudes between 65 and 4096. The index range is broken down into a number of segments, each of length 64. Amplitudes in the index range are encoded with a variable-length codeword and an index value, P . On the other hand, amplitudes in the base range do not require an index value.

For video sampled with a bit-depth of 10 or 12, P requires 6 bits. P requires 4 bits when 8-bit video sampling is used.

Coefficients with no preceding run of zeros are considered a different type of symbol from those that have one or more preceding zero-valued coefficients. If there are one or more preceding zero-valued coefficients the amplitude is encoded with a variable-length codeword (and, possibly an index value) and followed by another variable-length codeword representing the length of the preceding run of zeros. If there is no preceding run of zeros, the amplitude is encoded with a single variable-length codeword (and, possibly an index value). The codeword for a coefficient with a given amplitude and no preceding run of zeros is different from the codeword with the same amplitude and not having a preceding run of zeros. This enables the decoder to differentiate between the presence or absence of preceding zeros.

Annex B User Data Control and Payload Field Usage (Informative)

The UDL field and 260 bytes User Data Payload located at offset 0x60 in the header (see Table 8 and Sections 7.2.8 and 7.2.9) are used to respectively carry user data type-IDs and payloads. This standard does not specifically define data types or payload mappings. However, it does provide for future mappings using the 4 bit UDL field. Each bit of this field will be associated with a distinct data type. A UDL bit value of 0b will indicate that the defined data type is not present as payload data and a value of 1b will indicate that the corresponding data type is present.

As an example, carriage of SMPTE ST 291-1 Ancillary Data fields in the payload area could be defined as one of the four permissible payloads. It is anticipated that UDL mappings to distinct data types and their corresponding payload definitions will be done outside this document at some future date through the SMPTE standardization process.

Annex C Compression IDs (Normative)

The compression identifiers (CID) shall use the values described in the following table.

Table C.1 – HD compression parameters

CID	Source scan type	Samples per line	Active lines	Channel Sub-sampling	Bit depth	Quantization and VLC tables*	Compressed Frame Size (bytes)*	Coding Unit Size (bytes)	Compressed Payload Size (bytes)
1235	progressive	1920	1080 (frame)	4:2:2	10	D.1 E.1-E.3	917504	917504	916860
1237	progressive				8	D.2 E.4-E.6	606208	606208	605564
1238	progressive				8	D.3 E.7-E.9	917504	917504	916860
1241	interlaced	1920	540 (field)		10	D.4 E.1-E.3	917504	458752	458108
1242	interlaced				8	D.5 E.4-E.6	606208	303104	302460
1243	interlaced				8	D.6 E.7-E.9	917504	458752	458108
1244	interlaced	1440	540 (field)		8	D.7 E.4-E.6	606208	303104	302460
1250	progressive	1280	720 (frame)		10	D.8 E.10-E.12	458752	458752	458108
1251	progressive				8	D.9 E.13-E.15	458752	458752	458108
1252	progressive				8	D.10 E.16-E.18	303104	303104	302460
1253	progressive	1920	1080 (frame)		8	D.2 E.4-E.6	188416	188416	187772
1256	progressive	1920	1080 (frame)	RGB	10	D.11 E.1-E.3	1835008	1835008	1834364
1258	progressive	960	720 (frame)	4:2:2	8	D.10 E.16-E.18	212992	212992	212348
1259	progressive	1440	1080 (frame)		8	D.2 E.4-E.6	417792	417792	417148
1260	interlaced	1440	540 (field)		8	D.7 E.4-E.6	417792	417792	417148

*Note: The encoder will need to adjust the quantization scale factors qsf of the macroblocks (Section 8.2.2) to meet the required compressed frame size.

Table C.2 – RI compression parameters

CID	Source scan type	Channel Sub-sampling	Quantization and VLC tables	Bit depth	Reference Macroblock Count M_T	Reference compressed frame size (bytes) * C_0
1270	progressive	4:4:4(:4) RGB(A)	D.11 E.1-E.3	10 12	8160	1835008
1271	progressive	4:2:2(:4) 4:2:0(:4)	D.1 E.1-E.3	10 12	8160	917504
1272	progressive		D.3 E.7-E.9	8	8160	917504
1273	progressive		D.2 E.4-E.6	8	8160	606208
1274	progressive	4:2:2 4:2:0	D.2 E.4-E.6	8	8160	188416

* Note: The encoder will need to adjust the quantization scale factors qsf of the macroblocks (Section 8.2.2) to meet the required compressed frame size calculated by equ. (7.1) using the values of C_0 and M_T from this table.

Annex D Quantization Weighting Tables (Normative)

The compression identifiers noted here shall use the quantization weights in the following tables.

Table D.1 – Compression ID 1235, 1271

Luma Quantization Weighting Table							
-	32	32	32	33	35	38	39
32	33	32	33	36	36	39	42
32	32	33	36	35	37	41	43
31	33	34	36	36	40	42	48
32	34	36	37	39	42	46	51
36	37	37	39	41	46	51	55
37	39	41	41	47	50	55	56
41	42	41	44	50	53	60	60

Chroma Quantization Weighting Table							
-	32	33	34	39	41	54	59
33	34	35	38	43	49	58	84
34	37	39	44	46	55	74	87
40	42	47	48	58	70	87	86
43	50	56	63	72	94	91	82
55	63	65	75	93	89	85	73
61	67	82	81	83	90	79	73
74	84	75	78	90	85	73	73

Table D.2 – Compression ID 1237, 1253, 1259, 1273, 1274

Luma Quantization Weighting Table							
-	32	36	37	41	44	54	60
33	34	36	39	43	51	62	78
34	36	38	41	49	59	73	79
37	38	40	47	55	66	80	95
38	41	46	54	63	79	93	96
46	47	56	64	78	90	97	98
49	58	66	78	89	97	102	98
61	65	82	87	100	104	99	99

Chroma Quantization Weighting Table							
-	32	38	39	47	51	77	83
36	39	41	48	55	74	85	95
39	45	53	58	72	83	105	89
51	58	66	73	82	109	92	95
57	75	78	89	105	95	93	96
81	82	99	99	94	90	97	98
83	96	97	93	89	97	102	98
90	94	92	88	100	104	99	99

Table D.3 – Compression ID 1238, 1272

Luma Quantization Weighting Table							
-	32	33	33	37	37	41	42
32	34	33	35	36	38	41	43
33	33	33	36	37	39	41	44
33	33	35	36	38	40	44	50
33	36	35	38	39	44	49	51
38	36	38	39	44	47	51	58
38	39	41	44	47	51	57	59
41	42	45	47	53	55	57	57

Chroma Quantization Weighting Table							
-	32	34	34	40	41	55	61
35	35	35	39	42	52	62	75
35	39	41	44	52	58	68	77
43	45	48	53	58	65	74	80
45	55	59	63	66	74	76	89
59	65	66	70	77	73	90	84
63	66	74	76	73	82	86	82
65	70	73	73	77	80	82	82

Table D.4 – Compression ID 1241

Luma Quantization Weighting Table							
-	32	35	36	39	40	45	45
33	34	37	39	40	43	46	46
34	36	38	38	42	47	46	46
37	38	38	41	46	45	47	49
38	39	41	45	45	47	48	50
38	39	43	44	46	47	49	48
37	39	40	44	45	48	48	49
37	37	42	43	46	47	49	49

Chroma Quantization Weighting Table							
-	32	37	40	41	41	45	45
36	37	41	41	42	45	46	48
38	40	41	43	45	47	48	46
40	42	44	44	46	48	47	49
42	44	45	45	47	47	48	50
45	46	44	45	46	47	49	48
46	42	45	47	45	48	49	49
41	43	46	45	46	48	49	49

Table D.5 – Compression ID 1242

Luma Quantization Weighting Table							
-	32	35	36	38	37	40	42
33	34	35	37	37	37	45	49
33	33	37	37	36	46	48	50
33	36	36	37	44	46	52	51
35	37	39	41	48	52	51	50
37	38	42	50	50	50	49	49
37	44	49	49	50	49	51	47
45	47	47	48	51	52	47	47

Chroma Quantization Weighting Table							
-	32	45	45	51	47	45	47
37	45	44	49	41	47	50	51
42	38	44	40	47	51	49	51
37	42	43	51	49	48	52	54
40	44	54	46	51	52	54	53
46	51	47	55	54	53	53	60
48	49	55	54	52	55	62	60
47	50	49	49	59	63	60	60

Table D.6 – Compression ID 1243

Luma Quantization Weighting Table							
-	32	35	35	38	40	44	45
32	33	35	35	39	42	44	45
33	35	34	37	41	42	45	45
35	35	37	40	41	44	45	48
35	37	38	40	43	45	47	48
36	36	38	40	45	47	48	47
35	36	38	41	45	46	47	48
36	37	39	41	44	45	47	47

Chroma Quantization Weighting Table							
-	32	37	39	41	42	45	45
36	36	39	41	43	45	46	45
37	41	41	43	45	44	45	46
43	42	43	46	44	45	46	48
43	44	47	45	44	46	47	49
44	46	44	45	45	47	48	47
44	42	46	44	45	46	47	48
41	43	45	44	45	46	47	47

Table D.7 – Compression IDs 1244 and 1260

Luma Quantization Weighting Table							
--	32	37	37	40	41	52	53
33	36	36	38	40	48	49	52
34	34	37	39	44	47	49	54
33	35	38	40	45	46	54	51
34	37	37	42	44	49	52	48
34	34	38	43	44	51	50	50
33	36	41	44	51	52	50	54
36	38	44	47	53	53	54	54

Chroma Quantization Weighting Table							
--	32	40	38	42	40	45	45
34	42	36	43	38	46	46	49
38	35	43	39	44	47	47	49
35	42	43	42	46	47	49	52
38	43	43	44	50	49	56	50
42	43	44	50	51	57	52	53
41	45	46	53	53	56	53	54
46	46	51	49	56	53	58	58

Table D.8 – Compression ID 1250

Luma Quantization Weighting Table							
-	32	35	35	36	36	41	43
32	34	35	36	37	39	43	47
33	34	36	38	38	42	42	50
34	36	38	38	41	40	47	54
35	38	39	40	39	45	49	58
38	39	40	39	46	47	54	60
38	39	41	46	46	48	57	62
40	41	44	45	49	54	63	63

Chroma Quantization Weighting Table							
-	32	35	36	40	42	51	51
35	36	39	39	43	51	52	55
36	41	41	43	51	53	54	56
43	44	45	50	54	54	55	57
45	48	50	51	55	58	59	58
49	52	49	57	58	62	58	60
51	51	56	58	62	61	59	62
52	52	60	61	59	59	63	63

Table D.9 – Compression ID 1251

Luma Quantization Weighting Table							
-	32	34	34	35	36	41	44
32	34	35	36	38	39	43	46
34	35	36	38	38	41	43	46
35	37	38	38	40	40	44	55
36	38	38	40	39	44	50	58
38	38	40	40	44	48	53	61
38	40	41	42	46	48	58	62
39	40	41	43	50	55	62	62

Chroma Quantization Weighting Table							
-	32	35	36	40	42	51	51
35	36	39	39	43	51	52	55
36	41	41	43	51	53	54	56
43	44	45	50	54	54	55	57
45	48	50	51	55	58	59	58
48	50	49	57	58	62	58	61
48	51	56	58	62	61	59	62
52	52	60	61	59	59	62	62

Table D.10 – Compression ID 1252, 1258

Luma Quantization Weighting Table							
-	32	36	36	40	40	55	60
34	36	37	40	41	48	57	82
35	36	41	41	46	52	73	82
37	40	42	45	50	65	80	87
39	41	44	49	62	78	88	90
41	44	49	58	73	90	95	95
43	52	55	68	90	100	97	93
52	53	71	82	107	103	99	99

Chroma Quantization Weighting Table							
-	32	37	38	49	53	65	66
35	37	40	49	56	64	65	82
36	42	50	56	64	67	73	85
46	50	57	63	71	72	89	87
49	58	65	72	78	88	88	90
60	64	74	81	84	90	95	134
62	74	77	80	90	114	129	125
74	74	90	100	128	125	116	116

Table D.11 – Compression ID 1256 and 1270 (Video)
Compression ID 1271, 1272, 1273 (Alpha)

Quantization Weighting Table							
	32	32	32	33	35	38	39
32	33	32	33	36	36	39	42
32	32	33	36	35	37	41	43
31	33	34	36	36	40	42	48
32	34	36	37	39	42	46	51
36	37	37	39	41	46	51	55
37	39	41	41	47	50	55	56
41	42	41	44	50	53	60	60

Annex E VLC Tables (Normative)

The compression identifiers noted here shall use the parameters described in the following tables. Codewords shown are binary values in the format *x*b, with the msb to the left and the 'b' dropped.

Compression ID 1235, 1241, 1256, 1270 and 1271

Table E.1 – Amplitude VLC Table for Compression ID 1235, 1241, 1256, 1270 and 1271

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
00	2	1	0	0
01	2	1	1	0
100	3	2	0	0
1010	4	3	0	0
1011	4	EOB[1]	0	0
11000	5	4	0	0
11001	5	5	0	0
11010	5	2	1	0
110110	6	6	0	0
110111	6	7	0	0
111000	6	8	0	0
111001	6	3	1	0
1110100	7	9	0	0
1110101	7	10	0	0
1110110	7	11	0	0
1110111	7	4	1	0
11110000	8	12	0	0
11110001	8	13	0	0
11110010	8	14	0	0
11110011	8	15	0	0
11110100	8	16	0	0
11110101	8	5	1	0
111101100	9	17	0	0
111101101	9	18	0	0
111101110	9	19	0	0
111101111	9	20	0	0
111110000	9	21	0	0
111110001	9	6	1	0
111110010	9	7	1	0
1111100110	10	22	0	0
1111100111	10	23	0	0
1111101000	10	24	0	0
1111101001	10	25	0	0
1111101010	10	26	0	0
1111101011	10	27	0	0
1111101100	10	28	0	0
1111101101	10	29	0	0
1111101110	10	8	1	0
1111101111	10	9	1	0
11111100000	11	30	0	0
11111100001	11	31	0	0
11111100010	11	32	0	0
11111100011	11	33	0	0
11111100100	11	34	0	0
11111100101	11	35	0	0
11111100110	11	36	0	0
11111100111	11	37	0	0
11111101000	11	38	0	0
11111101001	11	10	1	0
11111101010	11	11	1	0
111111010110	12	39	0	0
111111010111	12	40	0	0
111111011000	12	41	0	0
111111011001	12	42	0	0
111111011010	12	43	0	0
111111011011	12	44	0	0
111111011100	12	45	0	0
111111011101	12	46	0	0
111111011110	12	47	0	0
111111011111	12	48	0	0
111111100000	12	49	0	0
111111100001	12	50	0	0
111111100010	12	12	1	0
111111100011	12	13	1	0
111111100100	12	14	1	0
111111100101	12	15	1	0
1111111001100	13	51	0	0
1111111001101	13	52	0	0
1111111001110	13	53	0	0
1111111001111	13	54	0	0
1111111010000	13	55	0	0
1111111010001	13	56	0	0
1111111010010	13	57	0	0
1111111010011	13	58	0	0
1111111010100	13	59	0	0
1111111010101	13	60	0	0
1111111010110	13	61	0	0
1111111010111	13	62	0	0
1111111011000	13	63	0	0
1111111011001	13	1	0	1
1111111011010	13	16	1	0
1111111011011	13	17	1	0
1111111011100	13	18	1	0
1111111011101	13	19	1	0
11111110111100	14	64	0	0
11111110111101	14	2	0	1
11111110111110	14	3	0	1
11111110111111	14	4	0	1
11111111000000	14	5	0	1
11111111000001	14	6	0	1
11111111000010	14	7	0	1
11111111000011	14	8	0	1

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
11111111000100	14	9	0	1
11111111000101	14	10	0	1
11111111000110	14	11	0	1
11111111000111	14	12	0	1
11111111001000	14	13	0	1
11111111001001	14	14	0	1
11111111001010	14	15	0	1
11111111001011	14	16	0	1
11111111001100	14	17	0	1
11111111001101	14	20	1	0
11111111001110	14	21	1	0
11111111001111	14	22	1	0
11111111010000	14	23	1	0
11111111010001	14	24	1	0
111111110100100	15	18	0	1
111111110100101	15	19	0	1
111111110100110	15	20	0	1
111111110100111	15	21	0	1
111111110101000	15	22	0	1
111111110101001	15	23	0	1
111111110101010	15	24	0	1
111111110101011	15	25	0	1
111111110101100	15	26	0	1
111111110101101	15	27	0	1
111111110101110	15	28	0	1
111111110101111	15	29	0	1
111111110110000	15	30	0	1
111111110110001	15	31	0	1
111111110110010	15	32	0	1
111111110110011	15	33	0	1
111111110110100	15	34	0	1
111111110110101	15	35	0	1
111111110110110	15	36	0	1
111111110110111	15	37	0	1
111111110111000	15	38	0	1
111111110111001	15	39	0	1
111111110111010	15	40	0	1
111111110111011	15	41	0	1
111111110111100	15	42	0	1
111111110111101	15	25	1	0
111111110111110	15	26	1	0
111111110111111	15	27	1	0
111111111000000	15	28	1	0
111111111000001	15	29	1	0
111111111000010	15	30	1	0
111111111000011	15	31	1	0
111111111000100	15	32	1	0
1111111110001010	16	43	0	1
1111111110001011	16	44	0	1
1111111110001100	16	45	0	1
1111111110001101	16	46	0	1
1111111110001110	16	47	0	1
1111111110001111	16	48	0	1
1111111110010000	16	49	0	1
1111111110010001	16	50	0	1
1111111110010010	16	51	0	1
1111111110010011	16	52	0	1
1111111110010100	16	53	0	1

Codeword	Length	Amp	Run Flag	Index Flag
1111111110010101	16	54	0	1
1111111110010110	16	55	0	1
1111111110010111	16	56	0	1
1111111110011000	16	57	0	1
1111111110011001	16	58	0	1
1111111110011010	16	59	0	1
1111111110011011	16	60	0	1
1111111110011100	16	61	0	1
1111111110011101	16	62	0	1
1111111110011110	16	63	0	1
1111111110011111	16	64	0	1
1111111110100000	16	33	1	0
1111111110100001	16	34	1	0
1111111110100010	16	35	1	0
1111111110100011	16	36	1	0
1111111110100100	16	37	1	0
1111111110100101	16	38	1	0
1111111110100110	16	39	1	0
1111111110100111	16	40	1	0
1111111110101000	16	41	1	0
1111111110101001	16	42	1	0
1111111110101010	16	43	1	0
1111111110101011	16	44	1	0
1111111110101100	16	45	1	0
1111111110101101	16	46	1	0
1111111110101110	16	47	1	0
1111111110101111	16	48	1	0
1111111110110000	16	49	1	0
1111111110110001	16	50	1	0
1111111110110010	16	51	1	0
1111111110110011	16	52	1	0
1111111110110100	16	53	1	0
1111111110110101	16	54	1	0
1111111110110110	16	55	1	0
1111111110110111	16	56	1	0
1111111110111000	16	57	1	0
1111111110111001	16	58	1	0
1111111110111010	16	59	1	0
1111111110111011	16	60	1	0
1111111110111100	16	61	1	0
1111111110111101	16	62	1	0
1111111110111110	16	63	1	0
1111111110111111	16	64	1	0
1111111111000000	16	1	1	1
1111111111000001	16	2	1	1
1111111111000010	16	3	1	1
1111111111000011	16	4	1	1
1111111111000100	16	5	1	1
1111111111000101	16	6	1	1
1111111111000110	16	7	1	1
1111111111000111	16	8	1	1
1111111111001000	16	9	1	1
1111111111001001	16	10	1	1
1111111111001010	16	11	1	1
1111111111001011	16	12	1	1
1111111111001100	16	13	1	1
1111111111001101	16	14	1	1
1111111111001110	16	15	1	1

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
1111111111001111	16	16	1	1
1111111111010000	16	17	1	1
1111111111010001	16	18	1	1
1111111111010010	16	19	1	1
1111111111010011	16	20	1	1
1111111111010100	16	21	1	1
1111111111010101	16	22	1	1
1111111111010110	16	23	1	1
1111111111010111	16	24	1	1
1111111111011000	16	25	1	1
1111111111011001	16	26	1	1
1111111111011010	16	27	1	1
1111111111011011	16	28	1	1
1111111111011100	16	29	1	1
1111111111011101	16	30	1	1
1111111111011110	16	31	1	1
1111111111011111	16	32	1	1
1111111111100000	16	33	1	1
1111111111100001	16	34	1	1
1111111111100010	16	35	1	1
1111111111100011	16	36	1	1
1111111111100100	16	37	1	1
1111111111100101	16	38	1	1
1111111111100110	16	39	1	1
1111111111100111	16	40	1	1
1111111111101000	16	41	1	1
1111111111101001	16	42	1	1
1111111111101010	16	43	1	1
1111111111101011	16	44	1	1
1111111111101100	16	45	1	1
1111111111101101	16	46	1	1
1111111111101110	16	47	1	1
1111111111101111	16	48	1	1
1111111111110000	16	49	1	1
1111111111110001	16	50	1	1
1111111111110010	16	51	1	1
1111111111110011	16	52	1	1
1111111111110100	16	53	1	1
1111111111110101	16	54	1	1
1111111111110110	16	55	1	1
1111111111110111	16	56	1	1
1111111111111000	16	57	1	1
1111111111111001	16	58	1	1
1111111111111010	16	59	1	1
1111111111111011	16	60	1	1
1111111111111100	16	61	1	1
1111111111111101	16	62	1	1
1111111111111110	16	63	1	1
1111111111111111	16	64	1	1

[1] See Section 7.3.1.1 for the definition of the End-Of-Block (EOB) codeword

Table E.2 – Run VLC Table for Compression IDs 1235, 1241, 1256, 1270 and 1271

Codeword	Length	Run
0	1	1
100	3	2
1010	4	3
1011	4	4
11000	5	5
11001	5	6
11010	5	7
11011	5	8
111000	6	9
111001	6	10
111010	6	11
111011	6	12
1111000	7	13
11110010	8	14
111100110	9	15
111100111	9	16
111101000	9	18
111101001	9	20
1111010100	10	17
1111010101	10	19
1111010110	10	21
1111010111	10	22
1111011000	10	23
1111011001	10	24
1111011010	10	25
1111011011	10	26
1111011100	10	27
1111011101	10	28
1111011110	10	29
1111011111	10	30
1111100000	10	31

Codeword	Length	Run
1111100001	10	32
1111100010	10	33
1111100011	10	34
1111100100	10	35
1111100101	10	36
1111100110	10	37
1111100111	10	38
1111101000	10	39
1111101001	10	40
1111101010	10	41
1111101011	10	42
1111101100	10	43
1111101101	10	44
1111101110	10	45
1111101111	10	46
1111110000	10	47
1111110001	10	48
1111110010	10	49
1111110011	10	50
1111110100	10	51
1111110101	10	52
1111110110	10	53
1111110111	10	54
1111111000	10	55
1111111001	10	56
1111111010	10	57
1111111011	10	58
1111111100	10	59
1111111101	10	60
1111111110	10	61
1111111111	10	62

Table E.3 – DC VLC Table for Compression IDs 1235, 1241, 1256, 1270 and 1271

Codeword	η
1010	0
111110	1
1011	2
1100	3
1101	4
000	5
001	6
010	7
011	8
100	9
1110	10
11110	11
1111110	12
1111111	13

VCL Tables for Compression ID 1237, 1242, 1244, 1253, 1259, 1260, 1273, 1274

Table E.4 – Amplitude VLC Table for Compression ID 1237, 1242, 1244, 1253, 1259, 1260, 1273, 1274

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
00	2	1	0	0
01	2	1	1	0
100	3	2	0	0
101	3	EOB[1]	0	0
1100	4	3	0	0
11010	5	4	0	0
11011	5	2	1	0
111000	6	5	0	0
111001	6	6	0	0
111010	6	7	0	0
111011	6	3	1	0
1111000	7	8	0	0
1111001	7	9	0	0
11110100	8	10	0	0
11110101	8	11	0	0
11110110	8	12	0	0
11110111	8	4	1	0
11111000	8	5	1	0
111110010	9	13	0	0
111110011	9	14	0	0
111110100	9	15	0	0
111110101	9	16	0	0
111110110	9	6	1	0
1111101110	10	17	0	0
1111101111	10	18	0	0
1111110000	10	19	0	0
1111110001	10	20	0	0
1111110010	10	21	0	0
1111110011	10	7	1	0
11111101000	11	22	0	0
11111101001	11	23	0	0
11111101010	11	24	0	0
11111101011	11	25	0	0
11111101100	11	26	0	0
11111101101	11	27	0	0
11111101110	11	8	1	0
11111101111	11	9	1	0
111111100000	12	28	0	0
111111100001	12	29	0	0
111111100010	12	30	0	0
111111100011	12	31	0	0
111111100100	12	32	0	0
111111100101	12	33	0	0
111111100110	12	34	0	0
111111100111	12	10	1	0
111111101000	12	11	1	0
111111101001	12	12	1	0
1111111010100	13	35	0	0
1111111010101	13	36	0	0
1111111010110	13	37	0	0
1111111010111	13	38	0	0
1111111011000	13	39	0	0
1111111011001	13	40	0	0
1111111011010	13	41	0	0
1111111011011	13	13	1	0
1111111011100	13	14	1	0

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
1111111011101	13	15	1	0
1111111011110	13	16	1	0
11111110111110	14	42	0	0
11111110111111	14	43	0	0
11111111000000	14	44	0	0
11111111000001	14	45	0	0
11111111000010	14	46	0	0
11111111000011	14	47	0	0
11111111000100	14	48	0	0
11111111000101	14	49	0	0
11111111000110	14	50	0	0
11111111000111	14	51	0	0
11111111001000	14	52	0	0
11111111001001	14	17	1	0
11111111001010	14	18	1	0
11111111001011	14	19	1	0
11111111001100	14	20	1	0
11111111001101	14	21	1	0
111111110011100	15	53	0	0
111111110011101	15	54	0	0
111111110011110	15	55	0	0
111111110011111	15	56	0	0
111111110100000	15	57	0	0
111111110100001	15	58	0	0
111111110100010	15	59	0	0
111111110100011	15	60	0	0
111111110100100	15	61	0	0
111111110100101	15	64	0	0
111111110100110	15	1	0	1
111111110100111	15	22	1	0
111111110101000	15	23	1	0
111111110101001	15	24	1	0
111111110101010	15	25	1	0
111111110101011	15	26	1	0
111111110101100	15	27	1	0
1111111101011010	16	62	0	0
1111111101011011	16	63	0	0
1111111101011100	16	2	0	1
1111111101011101	16	3	0	1
1111111101011110	16	4	0	1
1111111101011111	16	5	0	1
1111111101100000	16	6	0	1
1111111101100001	16	7	0	1
1111111101100010	16	8	0	1
1111111101100011	16	9	0	1
1111111101100100	16	10	0	1
1111111101100101	16	11	0	1
1111111101100110	16	12	0	1
1111111101100111	16	13	0	1
1111111101101000	16	14	0	1
1111111101101001	16	15	0	1
1111111101101010	16	16	0	1
1111111101101011	16	17	0	1
1111111101101100	16	18	0	1
1111111101101101	16	19	0	1
1111111101101110	16	20	0	1

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
1111111101101111	16	21	0	1
1111111101110000	16	22	0	1
1111111101110001	16	23	0	1
1111111101110010	16	24	0	1
1111111101110011	16	25	0	1
1111111101110100	16	26	0	1
1111111101110101	16	27	0	1
1111111101110110	16	28	0	1
1111111101110111	16	29	0	1
1111111101111000	16	30	0	1
1111111101111001	16	31	0	1
1111111101111010	16	32	0	1
1111111101111011	16	33	0	1
1111111101111100	16	34	0	1
1111111101111101	16	35	0	1
1111111101111110	16	36	0	1
1111111101111111	16	37	0	1
1111111110000000	16	38	0	1
1111111110000001	16	39	0	1
1111111110000010	16	40	0	1
1111111110000011	16	41	0	1
1111111110000100	16	42	0	1
1111111110000101	16	43	0	1
1111111110000110	16	44	0	1
1111111110000111	16	45	0	1
1111111110001000	16	46	0	1
1111111110001001	16	47	0	1
1111111110001010	16	48	0	1
1111111110001011	16	49	0	1
1111111110001100	16	50	0	1
1111111110001101	16	51	0	1
1111111110001110	16	52	0	1
1111111110001111	16	53	0	1
1111111110010000	16	54	0	1
1111111110010001	16	55	0	1
1111111110010010	16	56	0	1
1111111110010011	16	57	0	1
1111111110010100	16	58	0	1
1111111110010101	16	59	0	1
1111111110010110	16	60	0	1
1111111110010111	16	61	0	1
1111111110011000	16	62	0	1
1111111110011001	16	63	0	1
1111111110011010	16	64	0	1
1111111110011011	16	28	1	0
1111111110011100	16	29	1	0
1111111110011101	16	30	1	0
1111111110011110	16	31	1	0
1111111110011111	16	32	1	0
1111111110100000	16	33	1	0
1111111110100001	16	34	1	0
1111111110100010	16	35	1	0
1111111110100011	16	36	1	0
1111111110100100	16	37	1	0
1111111110100101	16	38	1	0
1111111110100110	16	39	1	0
1111111110100111	16	40	1	0
1111111110101000	16	41	1	0
1111111110101001	16	42	1	0
1111111110101010	16	43	1	0
1111111110101011	16	44	1	0
1111111110101100	16	45	1	0

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
1111111110101101	16	46	1	0
1111111110101110	16	47	1	0
1111111110101111	16	48	1	0
1111111110110000	16	49	1	0
1111111110110001	16	50	1	0
1111111110110010	16	51	1	0
1111111110110011	16	52	1	0
1111111110110100	16	53	1	0
1111111110110101	16	54	1	0
1111111110110110	16	55	1	0
1111111110110111	16	56	1	0
1111111110111000	16	57	1	0
1111111110111001	16	58	1	0
1111111110111010	16	59	1	0
1111111110111011	16	60	1	0
1111111110111100	16	61	1	0
1111111110111101	16	62	1	0
1111111110111110	16	63	1	0
1111111110111111	16	64	1	0
1111111111000000	16	1	1	1
1111111111000001	16	2	1	1
1111111111000010	16	3	1	1
1111111111000011	16	4	1	1
1111111111000100	16	5	1	1
1111111111000101	16	6	1	1
1111111111000110	16	7	1	1
1111111111000111	16	8	1	1
1111111111001000	16	9	1	1
1111111111001001	16	10	1	1
1111111111001010	16	11	1	1
1111111111001011	16	12	1	1
1111111111001100	16	13	1	1
1111111111001101	16	14	1	1
1111111111001110	16	15	1	1
1111111111001111	16	16	1	1
1111111111010000	16	17	1	1
1111111111010001	16	18	1	1
1111111111010010	16	19	1	1
1111111111010011	16	20	1	1
1111111111010100	16	21	1	1
1111111111010101	16	22	1	1
1111111111010110	16	23	1	1
1111111111010111	16	24	1	1
1111111111011000	16	25	1	1
1111111111011001	16	26	1	1
1111111111011010	16	27	1	1
1111111111011011	16	28	1	1
1111111111011100	16	29	1	1
1111111111011101	16	30	1	1
1111111111011110	16	31	1	1
1111111111011111	16	32	1	1
1111111111100000	16	33	1	1
1111111111100001	16	34	1	1
1111111111100010	16	35	1	1
1111111111100011	16	36	1	1
1111111111100100	16	37	1	1
1111111111100101	16	38	1	1
1111111111100110	16	39	1	1
1111111111100111	16	40	1	1
1111111111101000	16	41	1	1
1111111111101001	16	42	1	1
1111111111101010	16	43	1	1

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
1111111111101011	16	44	1	1
1111111111101100	16	45	1	1
1111111111101101	16	46	1	1
1111111111101110	16	47	1	1
1111111111101111	16	48	1	1
1111111111110000	16	49	1	1
1111111111110001	16	50	1	1
1111111111110010	16	51	1	1
1111111111110011	16	52	1	1
1111111111110100	16	53	1	1
1111111111110101	16	54	1	1

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
1111111111110110	16	55	1	1
1111111111110111	16	56	1	1
1111111111111000	16	57	1	1
1111111111111001	16	58	1	1
1111111111111010	16	59	1	1
1111111111111011	16	60	1	1
1111111111111100	16	61	1	1
1111111111111101	16	62	1	1
1111111111111110	16	63	1	1
1111111111111111	16	64	1	1

[1] See Section 7.3.1.1 for the definition of the End-Of-Block (EOB) codeword

Table E.5 – Run VLC Table for Compression ID 1237, 1242, 1244, 1253, 1259, 1260, 1273,1274

Codeword	Length	Run
0	1	1
100	3	2
1010	4	3
1011	4	4
11000	5	5
11001	5	6
11010	5	7
110110	6	8
110111	6	9
111000	6	10
111001	6	11
111010	6	12
1110110	7	13
1110111	7	14
11110000	8	15
111100010	9	16
111100011	9	17
111100100	9	18
111100101	9	19
111100110	9	20
111100111	9	21
111101000	9	53
111101001	9	57
111101010	9	58
111101011	9	59
111101100	9	60
111101101	9	61
111101110	9	62
1111011110	10	22
1111011111	10	23
1111100000	10	24

Codeword	Length	Run
1111100001	10	25
1111100010	10	26
1111100011	10	27
1111100100	10	28
1111100101	10	29
1111100110	10	30
1111100111	10	31
1111101000	10	32
1111101001	10	33
1111101010	10	34
1111101011	10	35
1111101100	10	36
1111101101	10	37
1111101110	10	38
1111101111	10	39
1111110000	10	40
1111110001	10	41
1111110010	10	42
1111110011	10	43
1111110100	10	44
1111110101	10	45
1111110110	10	46
1111110111	10	47
1111111000	10	48
1111111001	10	49
1111111010	10	50
1111111011	10	51
1111111100	10	52
1111111101	10	54
1111111110	10	55
1111111111	10	56

Table E.6 – DC Differential VLC Table for Compression ID 1237, 1242, 1244, 1253, 1259, 1260, 1273, 1274

Codeword	η
000	0
1100	1
1101	2
001	3
010	4
011	5
100	6
101	7
1110	8
11110	9
111110	10
111111	11

VCL Tables for Compression ID 1238, 1243 and 1272

Table E.7 – Amplitude VLC Table for Compression ID 1238, 1243 and 1272

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
00	2	1	0	0
01	2	1	1	0
100	3	2	0	0
1010	4	3	0	0
1011	4	EOB[1]	0	0
11000	5	4	0	0
11001	5	5	0	0
11010	5	2	1	0
110110	6	6	0	0
110111	6	7	0	0
111000	6	8	0	0
111001	6	3	1	0
1110100	7	9	0	0
1110101	7	10	0	0
1110110	7	11	0	0
1110111	7	4	1	0
11110000	8	12	0	0
11110001	8	13	0	0
11110010	8	14	0	0
11110011	8	15	0	0
11110100	8	16	0	0
11110101	8	5	1	0
111101100	9	17	0	0
111101101	9	18	0	0
111101110	9	19	0	0
111101111	9	20	0	0
111110000	9	21	0	0
111110001	9	22	0	0
111110010	9	6	1	0
111110011	9	7	1	0
1111101000	10	23	0	0
1111101001	10	24	0	0
1111101010	10	25	0	0
1111101011	10	26	0	0
1111101100	10	27	0	0
1111101101	10	28	0	0
1111101110	10	29	0	0
1111101111	10	8	1	0
1111110000	10	9	1	0
11111100010	11	30	0	0
11111100011	11	31	0	0
11111100100	11	32	0	0
11111100101	11	33	0	0
11111100110	11	34	0	0
11111100111	11	35	0	0
11111101000	11	36	0	0
11111101001	11	37	0	0
11111101010	11	10	1	0
11111101011	11	11	1	0
111111011000	12	38	0	0
111111011001	12	39	0	0
111111011010	12	40	0	0
111111011011	12	41	0	0
111111011100	12	42	0	0
111111011101	12	43	0	0
111111011110	12	44	0	0

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
111111011111	12	45	0	0
111111100000	12	46	0	0
111111100001	12	47	0	0
111111100010	12	48	0	0
111111100011	12	12	1	0
111111100100	12	13	1	0
111111100101	12	14	1	0
1111111001100	13	49	0	0
1111111001101	13	50	0	0
1111111001110	13	51	0	0
1111111001111	13	52	0	0
1111111010000	13	53	0	0
1111111010001	13	54	0	0
1111111010010	13	55	0	0
1111111010011	13	56	0	0
1111111010100	13	57	0	0
1111111010101	13	58	0	0
1111111010110	13	59	0	0
1111111010111	13	60	0	0
1111111011000	13	61	0	0
1111111011001	13	15	1	0
1111111011010	13	16	1	0
1111111011011	13	17	1	0
1111111011100	13	18	1	0
11111110111010	14	62	0	0
11111110111011	14	63	0	0
11111110111100	14	64	0	0
11111110111101	14	1	0	1
11111110111110	14	2	0	1
11111110111111	14	3	0	1
11111111000000	14	4	0	1
11111111000001	14	5	0	1
11111111000010	14	6	0	1
11111111000011	14	7	0	1
11111111000100	14	8	0	1
11111111000101	14	9	0	1
11111111000110	14	10	0	1
11111111000111	14	11	0	1
11111111001000	14	12	0	1
11111111001001	14	13	0	1
11111111001010	14	14	0	1
11111111001011	14	15	0	1
11111111001100	14	16	0	1
11111111001101	14	19	1	0
11111111001110	14	20	1	0
11111111001111	14	21	1	0
11111111010000	14	22	1	0
11111111010001	14	23	1	0
11111111010010	14	24	1	0
111111110100110	15	17	0	1
111111110100111	15	18	0	1
111111110101000	15	19	0	1
111111110101001	15	20	0	1
111111110101010	15	21	0	1
111111110101011	15	22	0	1
111111110101100	15	23	0	1

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
111111110101101	15	24	0	1
111111110101110	15	25	0	1
111111110101111	15	26	0	1
111111110110000	15	27	0	1
111111110110001	15	28	0	1
111111110110010	15	29	0	1
111111110110011	15	30	0	1
111111110110100	15	31	0	1
111111110110101	15	32	0	1
111111110110110	15	33	0	1
111111110110111	15	34	0	1
111111110111000	15	35	0	1
111111110111001	15	36	0	1
111111110111010	15	37	0	1
111111110111011	15	40	0	1
111111110111100	15	25	1	0
111111110111101	15	26	1	0
111111110111110	15	27	1	0
111111110111111	15	28	1	0
111111111000000	15	29	1	0
111111111000001	15	30	1	0
1111111110000100	16	38	0	1
1111111110000101	16	39	0	1
1111111110000110	16	41	0	1
1111111110000111	16	42	0	1
1111111110001000	16	43	0	1
1111111110001001	16	44	0	1
1111111110001010	16	45	0	1
1111111110001011	16	46	0	1
1111111110001100	16	47	0	1
1111111110001101	16	48	0	1
1111111110001110	16	49	0	1
1111111110001111	16	50	0	1
1111111110010000	16	51	0	1
1111111110010001	16	52	0	1
1111111110010010	16	53	0	1
1111111110010011	16	54	0	1
1111111110010100	16	55	0	1
1111111110010101	16	56	0	1
1111111110010110	16	57	0	1
1111111110010111	16	58	0	1
1111111110011000	16	59	0	1
1111111110011001	16	60	0	1
1111111110011010	16	61	0	1
1111111110011011	16	62	0	1
1111111110011100	16	63	0	1
1111111110011101	16	64	0	1
1111111110011110	16	31	1	0
1111111110011111	16	32	1	0
1111111110100000	16	33	1	0
1111111110100001	16	34	1	0
1111111110100010	16	35	1	0
1111111110100011	16	36	1	0
1111111110100100	16	37	1	0
1111111110100101	16	38	1	0
1111111110100110	16	39	1	0
1111111110100111	16	40	1	0
1111111110101000	16	41	1	0
1111111110101001	16	42	1	0
1111111110101010	16	43	1	0
1111111110101011	16	44	1	0
1111111110101100	16	45	1	0

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
1111111110101101	16	46	1	0
1111111110101110	16	47	1	0
1111111110101111	16	48	1	0
1111111110110000	16	49	1	0
1111111110110001	16	50	1	0
1111111110110010	16	51	1	0
1111111110110011	16	52	1	0
1111111110110100	16	53	1	0
1111111110110101	16	54	1	0
1111111110110110	16	55	1	0
1111111110110111	16	56	1	0
1111111110111000	16	57	1	0
1111111110111001	16	58	1	0
1111111110111010	16	59	1	0
1111111110111011	16	60	1	0
1111111110111100	16	61	1	0
1111111110111101	16	62	1	0
1111111110111110	16	63	1	0
1111111110111111	16	64	1	0
1111111111000000	16	1	1	1
1111111111000001	16	2	1	1
1111111111000010	16	3	1	1
1111111111000011	16	4	1	1
1111111111000100	16	5	1	1
1111111111000101	16	6	1	1
1111111111000110	16	7	1	1
1111111111000111	16	8	1	1
1111111111001000	16	9	1	1
1111111111001001	16	10	1	1
1111111111001010	16	11	1	1
1111111111001011	16	12	1	1
1111111111001100	16	13	1	1
1111111111001101	16	14	1	1
1111111111001110	16	15	1	1
1111111111001111	16	16	1	1
1111111111010000	16	17	1	1
1111111111010001	16	18	1	1
1111111111010010	16	19	1	1
1111111111010011	16	20	1	1
1111111111010100	16	21	1	1
1111111111010101	16	22	1	1
1111111111010110	16	23	1	1
1111111111010111	16	24	1	1
1111111111011000	16	25	1	1
1111111111011001	16	26	1	1
1111111111011010	16	27	1	1
1111111111011011	16	28	1	1
1111111111011100	16	29	1	1
1111111111011101	16	30	1	1
1111111111011110	16	31	1	1
1111111111011111	16	32	1	1
1111111111100000	16	33	1	1
1111111111100001	16	34	1	1
1111111111100010	16	35	1	1
1111111111100011	16	36	1	1
1111111111100100	16	37	1	1
1111111111100101	16	38	1	1
1111111111100110	16	39	1	1
1111111111100111	16	40	1	1
1111111111101000	16	41	1	1
1111111111101001	16	42	1	1
1111111111101010	16	43	1	1

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
1111111111101011	16	44	1	1
1111111111101100	16	45	1	1
1111111111101101	16	46	1	1
1111111111101110	16	47	1	1
1111111111101111	16	48	1	1
1111111111110000	16	49	1	1
1111111111110001	16	50	1	1
1111111111110010	16	51	1	1
1111111111110011	16	52	1	1
1111111111110100	16	53	1	1
1111111111110101	16	54	1	1

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
1111111111110110	16	55	1	1
1111111111110111	16	56	1	1
1111111111111000	16	57	1	1
1111111111111001	16	58	1	1
1111111111111010	16	59	1	1
1111111111111011	16	60	1	1
1111111111111100	16	61	1	1
1111111111111101	16	62	1	1
1111111111111110	16	63	1	1
1111111111111111	16	64	1	1

[1] See Section 7.3.1.1 for the definition of the End-Of-Block (EOB) codeword

Table E.8 – Run VLC Table for Compression ID 1238, 1243 and 1272

Codeword	Length	Run
0	1	1
100	3	2
1010	4	3
1011	4	4
11000	5	5
11001	5	6
11010	5	7
11011	5	8
111000	6	9
111001	6	10
111010	6	11
111011	6	12
1111000	7	13
11110010	8	14
111100110	9	15
111100111	9	16
111101000	9	20
111101001	9	21
1111010100	10	17
1111010101	10	18
1111010110	10	19
1111010111	10	22
1111011000	10	23
1111011001	10	24
1111011010	10	25
1111011011	10	26
1111011100	10	27
1111011101	10	28
1111011110	10	29
1111011111	10	30
1111100000	10	31

Codeword	Length	Run
1111100001	10	32
1111100010	10	33
1111100011	10	34
1111100100	10	35
1111100101	10	36
1111100110	10	37
1111100111	10	38
1111101000	10	39
1111101001	10	40
1111101010	10	41
1111101011	10	42
1111101100	10	43
1111101101	10	44
1111101110	10	45
1111101111	10	46
1111110000	10	47
1111110001	10	48
1111110010	10	49
1111110011	10	50
1111110100	10	51
1111110101	10	52
1111110110	10	53
1111110111	10	54
1111111000	10	55
1111111001	10	56
1111111010	10	57
1111111011	10	58
1111111100	10	59
1111111101	10	60
1111111110	10	61
1111111111	10	62

Table E.9 – DC Differential VLC Table for Compression ID 1238, 1243 and 1272

Codeword	η
000	0
1100	1
1101	2
001	3
010	4
011	5
100	6
101	7
1110	8
11110	9
111110	10
111111	11

VCL Tables for Compression ID 1250

Table E.10 – Amplitude VLC Table for Compression ID 1250

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
00	2	1	0	0
01	2	1	1	0
100	3	2	0	0
1010	4	3	0	0
1011	4	EOB[1]	0	0
11000	5	4	0	0
11001	5	5	0	0
11010	5	2	1	0
110110	6	6	0	0
110111	6	7	0	0
111000	6	8	0	0
111001	6	3	1	0
1110100	7	9	0	0
1110101	7	10	0	0
1110110	7	11	0	0
1110111	7	4	1	0
11110000	8	12	0	0
11110001	8	13	0	0
11110010	8	14	0	0
11110011	8	15	0	0
11110100	8	16	0	0
11110101	8	5	1	0
111101100	9	17	0	0
111101101	9	18	0	0
111101110	9	19	0	0
111101111	9	20	0	0
111110000	9	21	0	0
111110001	9	22	0	0
111110010	9	6	1	0
1111100110	10	23	0	0
1111100111	10	24	0	0
1111101000	10	25	0	0
1111101001	10	26	0	0
1111101010	10	27	0	0
1111101011	10	28	0	0
1111101100	10	29	0	0
1111101101	10	7	1	0
1111101110	10	8	1	0
11111011110	11	30	0	0
11111011111	11	31	0	0
11111100000	11	32	0	0
11111100001	11	33	0	0
11111100010	11	34	0	0
11111100011	11	35	0	0
11111100100	11	36	0	0
11111100101	11	37	0	0
11111100110	11	38	0	0
11111100111	11	39	0	0
11111101000	11	9	1	0
11111101001	11	10	1	0
111111010100	12	40	0	0
111111010101	12	41	0	0
111111010110	12	42	0	0
111111010111	12	43	0	0
111111011000	12	44	0	0
111111011001	12	45	0	0
111111011010	12	46	0	0

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
111111011011	12	47	0	0
111111011100	12	48	0	0
111111011101	12	49	0	0
111111011110	12	50	0	0
111111011111	12	51	0	0
111111100000	12	52	0	0
111111100001	12	11	1	0
111111100010	12	12	1	0
111111100011	12	13	1	0
1111111001000	13	53	0	0
1111111001001	13	54	0	0
1111111001010	13	55	0	0
1111111001011	13	56	0	0
1111111001100	13	57	0	0
1111111001101	13	58	0	0
1111111001110	13	59	0	0
1111111001111	13	60	0	0
1111111010000	13	61	0	0
1111111010001	13	62	0	0
1111111010010	13	63	0	0
1111111010011	13	64	0	0
1111111010100	13	1	0	1
1111111010101	13	2	0	1
1111111010110	13	3	0	1
1111111010111	13	4	0	1
1111111011000	13	5	0	1
1111111011001	13	14	1	0
1111111011010	13	15	1	0
1111111011011	13	16	1	0
1111111011100	13	17	1	0
11111110111010	14	6	0	1
11111110111011	14	7	0	1
11111110111100	14	8	0	1
11111110111101	14	9	0	1
11111110111110	14	10	0	1
11111110111111	14	11	0	1
11111111000000	14	12	0	1
11111111000001	14	13	0	1
11111111000010	14	14	0	1
11111111000011	14	15	0	1
11111111000100	14	16	0	1
11111111000101	14	17	0	1
11111111000110	14	18	0	1
11111111000111	14	19	0	1
11111111001000	14	20	0	1
11111111001001	14	21	0	1
11111111001010	14	22	0	1
11111111001011	14	23	0	1
11111111001100	14	24	0	1
11111111001101	14	25	0	1
11111111001110	14	26	0	1
11111111001111	14	18	1	0
11111111010000	14	19	1	0
11111111010001	14	20	1	0
11111111010010	14	21	1	0
111111110100110	15	27	0	1
111111110100111	15	28	0	1

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
111111110101000	15	29	0	1
111111110101001	15	30	0	1
111111110101010	15	31	0	1
111111110101011	15	32	0	1
111111110101100	15	33	0	1
111111110101101	15	34	0	1
111111110101110	15	35	0	1
111111110101111	15	36	0	1
111111110110000	15	37	0	1
111111110110001	15	38	0	1
111111110110010	15	39	0	1
111111110110011	15	40	0	1
111111110110100	15	41	0	1
111111110110101	15	42	0	1
111111110110110	15	43	0	1
111111110110111	15	44	0	1
111111110111000	15	45	0	1
111111110111001	15	46	0	1
111111110111010	15	47	0	1
111111110111011	15	48	0	1
111111110111100	15	49	0	1
111111110111101	15	50	0	1
111111110111110	15	51	0	1
111111110111111	15	52	0	1
111111111000000	15	53	0	1
111111111000001	15	55	0	1
111111111000010	15	56	0	1
111111111000011	15	22	1	0
111111111000100	15	23	1	0
111111111000101	15	24	1	0
111111111000110	15	25	1	0
111111111000111	15	26	1	0
111111111001000	15	27	1	0
1111111110010010	16	54	0	1
1111111110010011	16	57	0	1
1111111110010100	16	58	0	1
1111111110010101	16	59	0	1
1111111110010110	16	60	0	1
1111111110010111	16	61	0	1
1111111110011000	16	62	0	1
1111111110011001	16	63	0	1
1111111110011010	16	64	0	1
1111111110011011	16	28	1	0
1111111110011100	16	29	1	0
1111111110011101	16	30	1	0
1111111110011110	16	31	1	0
1111111110011111	16	32	1	0
1111111110100000	16	33	1	0
1111111110100001	16	34	1	0
1111111110100010	16	35	1	0
1111111110100011	16	36	1	0
1111111110100100	16	37	1	0
1111111110100101	16	38	1	0
1111111110100110	16	39	1	0
1111111110100111	16	40	1	0
1111111110101000	16	41	1	0
1111111110101001	16	42	1	0
1111111110101010	16	43	1	0
1111111110101011	16	44	1	0
1111111110101100	16	45	1	0
1111111110101101	16	46	1	0
1111111110101110	16	47	1	0

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
1111111110101111	16	48	1	0
1111111110110000	16	49	1	0
1111111110110001	16	50	1	0
1111111110110010	16	51	1	0
1111111110110011	16	52	1	0
1111111110110100	16	53	1	0
1111111110110101	16	54	1	0
1111111110110110	16	55	1	0
1111111110110111	16	56	1	0
1111111110111000	16	57	1	0
1111111110111001	16	58	1	0
1111111110111010	16	59	1	0
1111111110111011	16	60	1	0
1111111110111100	16	61	1	0
1111111110111101	16	62	1	0
1111111110111110	16	63	1	0
1111111110111111	16	64	1	0
1111111111000000	16	1	1	1
1111111111000001	16	2	1	1
1111111111000010	16	3	1	1
1111111111000011	16	4	1	1
1111111111000100	16	5	1	1
1111111111000101	16	6	1	1
1111111111000110	16	7	1	1
1111111111000111	16	8	1	1
1111111111001000	16	9	1	1
1111111111001001	16	10	1	1
1111111111001010	16	11	1	1
1111111111001011	16	12	1	1
1111111111001100	16	13	1	1
1111111111001101	16	14	1	1
1111111111001110	16	15	1	1
1111111111001111	16	16	1	1
1111111111010000	16	17	1	1
1111111111010001	16	18	1	1
1111111111010010	16	19	1	1
1111111111010011	16	20	1	1
1111111111010100	16	21	1	1
1111111111010101	16	22	1	1
1111111111010110	16	23	1	1
1111111111010111	16	24	1	1
1111111111011000	16	25	1	1
1111111111011001	16	26	1	1
1111111111011010	16	27	1	1
1111111111011011	16	28	1	1
1111111111011100	16	29	1	1
1111111111011101	16	30	1	1
1111111111011110	16	31	1	1
1111111111011111	16	32	1	1
1111111111100000	16	33	1	1
1111111111100001	16	34	1	1
1111111111100010	16	35	1	1
1111111111100011	16	36	1	1
1111111111100100	16	37	1	1
1111111111100101	16	38	1	1
1111111111100110	16	39	1	1
1111111111100111	16	40	1	1
1111111111101000	16	41	1	1
1111111111101001	16	42	1	1
1111111111101010	16	43	1	1
1111111111101011	16	44	1	1
1111111111101100	16	45	1	1

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
1111111111101101	16	46	1	1
1111111111101110	16	47	1	1
1111111111101111	16	48	1	1
1111111111110000	16	49	1	1
1111111111110001	16	50	1	1
1111111111110010	16	51	1	1
1111111111110011	16	52	1	1
1111111111110100	16	53	1	1
1111111111110101	16	54	1	1
1111111111110110	16	55	1	1

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
1111111111110111	16	56	1	1
1111111111111000	16	57	1	1
1111111111111001	16	58	1	1
1111111111111010	16	59	1	1
1111111111111011	16	60	1	1
1111111111111100	16	61	1	1
1111111111111101	16	62	1	1
1111111111111110	16	63	1	1
1111111111111111	16	64	1	1

[1] See Section 7.3.1.1 for the definition of the End-Of-Block (EOB) codeword

Table E.11 – Run VLC Table for Compression ID 1250

Codeword	Length	Run
0	1	1
100	3	2
101	3	3
1100	4	4
11010	5	5
11011	5	6
11100	5	7
111010	6	8
1110110	7	9
1110111	7	10
1111000	7	11
11110010	8	12
111100110	9	13
111100111	9	14
1111010000	10	15
1111010001	10	16
1111010010	10	17
1111010011	10	18
1111010100	10	19
1111010101	10	20
1111010110	10	21
1111010111	10	22
1111011000	10	23
1111011001	10	24
1111011010	10	25
1111011011	10	26
1111011100	10	27
1111011101	10	28
1111011110	10	29
1111011111	10	30
1111100000	10	31

Codeword	Length	Run
1111100001	10	32
1111100010	10	33
1111100011	10	34
1111100100	10	35
1111100101	10	36
1111100110	10	37
1111100111	10	38
1111101000	10	39
1111101001	10	40
1111101010	10	41
1111101011	10	42
1111101100	10	43
1111101101	10	44
1111101110	10	45
1111101111	10	46
1111110000	10	47
1111110001	10	48
1111110010	10	49
1111110011	10	50
1111110100	10	51
1111110101	10	52
1111110110	10	53
1111110111	10	54
1111111000	10	55
1111111001	10	56
1111111010	10	57
1111111011	10	58
1111111100	10	59
1111111101	10	60
1111111110	10	61
1111111111	10	62

Table E.12 – DC Differential VLC Table for Compression ID 1250

Codeword	η
1010	0
111110	1
1011	2
1100	3
1101	4
000	5
001	6
010	7
011	8
100	9
1110	10
11110	11
1111110	12
1111111	13

VLC Tables for Compression ID 1251

Table E.13 – Amplitude VLC Table for Compression ID 1251

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
00	2	1	0	0
01	2	1	1	0
100	3	2	0	0
1010	4	3	0	0
1011	4	EOB[1]	0	0
11000	5	4	0	0
11001	5	5	0	0
11010	5	2	1	0
110110	6	6	0	0
110111	6	7	0	0
111000	6	8	0	0
111001	6	3	1	0
1110100	7	9	0	0
1110101	7	10	0	0
1110110	7	11	0	0
1110111	7	4	1	0
11110000	8	12	0	0
11110001	8	13	0	0
11110010	8	14	0	0
11110011	8	15	0	0
11110100	8	16	0	0
11110101	8	5	1	0
111101100	9	17	0	0
111101101	9	18	0	0
111101110	9	19	0	0
111101111	9	20	0	0
111110000	9	21	0	0
111110001	9	6	1	0
1111100100	10	22	0	0
1111100101	10	23	0	0
1111100110	10	24	0	0
1111100111	10	25	0	0
1111101000	10	26	0	0
1111101001	10	27	0	0
1111101010	10	28	0	0
1111101011	10	29	0	0
1111101100	10	7	1	0
1111101101	10	8	1	0
11111011100	11	30	0	0
11111011101	11	31	0	0
11111011110	11	32	0	0
11111011111	11	33	0	0
11111100000	11	34	0	0
11111100001	11	35	0	0
11111100010	11	36	0	0
11111100011	11	37	0	0
11111100100	11	38	0	0
11111100101	11	39	0	0
11111100110	11	40	0	0
11111100111	11	9	1	0
11111101000	11	10	1	0
11111101001	11	11	1	0
1111111010100	12	41	0	0
1111111010101	12	42	0	0
1111111010110	12	43	0	0
1111111010111	12	44	0	0
1111111011000	12	45	0	0
1111111011001	12	46	0	0
1111111011010	12	47	0	0
1111111011011	12	48	0	0
1111111011100	12	49	0	0
1111111011101	12	50	0	0
1111111011110	12	51	0	0
1111111011111	12	52	0	0
1111111100000	12	12	1	0
1111111100001	12	13	1	0
1111111100010	12	14	1	0
1111111000110	13	53	0	0
1111111000111	13	54	0	0
1111111001000	13	55	0	0
1111111001001	13	56	0	0
1111111001010	13	57	0	0
1111111001011	13	58	0	0
1111111001100	13	59	0	0
1111111001101	13	60	0	0
1111111001110	13	61	0	0
1111111001111	13	62	0	0
1111111010000	13	63	0	0
1111111010001	13	64	0	0
1111111010010	13	1	0	1
1111111010011	13	2	0	1
1111111010100	13	3	0	1
1111111010101	13	4	0	1
1111111010110	13	5	0	1
1111111010111	13	6	0	1
1111111011000	13	7	0	1
1111111011001	13	8	0	1
1111111011010	13	15	1	0
1111111011011	13	16	1	0
1111111011100	13	17	1	0
11111110111010	14	9	0	1
11111110111011	14	10	0	1
11111110111100	14	11	0	1
11111110111101	14	12	0	1
11111110111110	14	13	0	1
11111110111111	14	14	0	1
11111111000000	14	15	0	1
11111111000001	14	16	0	1
11111111000010	14	17	0	1
11111111000011	14	18	0	1
11111111000100	14	19	0	1
11111111000101	14	20	0	1
11111111000110	14	21	0	1
11111111000111	14	22	0	1
11111111001000	14	23	0	1
11111111001001	14	24	0	1
11111111001010	14	25	0	1
11111111001011	14	26	0	1
11111111001100	14	27	0	1
11111111001101	14	28	0	1
11111111001110	14	29	0	1
11111111001111	14	18	1	0

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
11111111010000	14	19	1	0
11111111010001	14	20	1	0
11111111010010	14	21	1	0
11111111010011	14	22	1	0
111111110101000	15	30	0	1
111111110101001	15	31	0	1
111111110101010	15	32	0	1
111111110101011	15	33	0	1
111111110101100	15	34	0	1
111111110101101	15	35	0	1
111111110101110	15	36	0	1
111111110101111	15	37	0	1
111111110110000	15	38	0	1
111111110110001	15	39	0	1
111111110110010	15	40	0	1
111111110110011	15	41	0	1
111111110110100	15	42	0	1
111111110110101	15	43	0	1
111111110110110	15	44	0	1
111111110110111	15	45	0	1
111111110111000	15	46	0	1
111111110111001	15	47	0	1
111111110111010	15	48	0	1
111111110111011	15	49	0	1
111111110111100	15	50	0	1
111111110111101	15	51	0	1
111111110111110	15	52	0	1
111111110111111	15	53	0	1
111111111000000	15	54	0	1
111111111000001	15	55	0	1
111111111000010	15	56	0	1
111111111000011	15	57	0	1
111111111000100	15	58	0	1
111111111000101	15	23	1	0
111111111000110	15	24	1	0
111111111000111	15	25	1	0
111111111001000	15	26	1	0
111111111001001	15	27	1	0
111111111001010	15	28	1	0
1111111110010110	16	59	0	1
1111111110010111	16	60	0	1
1111111110011000	16	61	0	1
1111111110011001	16	62	0	1
1111111110011010	16	63	0	1
1111111110011011	16	64	0	1
1111111110011100	16	29	1	0
1111111110011101	16	30	1	0
1111111110011110	16	31	1	0
1111111110011111	16	32	1	0
1111111110100000	16	33	1	0
1111111110100001	16	34	1	0
1111111110100010	16	35	1	0
1111111110100011	16	36	1	0
1111111110100100	16	37	1	0
1111111110100101	16	38	1	0
1111111110100110	16	39	1	0
1111111110100111	16	40	1	0
1111111110101000	16	41	1	0
1111111110101001	16	42	1	0
1111111110101010	16	43	1	0
1111111110101011	16	44	1	0
1111111110101100	16	45	1	0

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
1111111110101101	16	46	1	0
1111111110101110	16	47	1	0
1111111110101111	16	48	1	0
1111111110110000	16	49	1	0
1111111110110001	16	50	1	0
1111111110110010	16	51	1	0
1111111110110011	16	52	1	0
1111111110110100	16	53	1	0
1111111110110101	16	54	1	0
1111111110110110	16	55	1	0
1111111110110111	16	56	1	0
1111111110111000	16	57	1	0
1111111110111001	16	58	1	0
1111111110111010	16	59	1	0
1111111110111011	16	60	1	0
1111111110111100	16	61	1	0
1111111110111101	16	62	1	0
1111111110111110	16	63	1	0
1111111110111111	16	64	1	0
1111111111000000	16	1	1	1
1111111111000001	16	2	1	1
1111111111000010	16	3	1	1
1111111111000011	16	4	1	1
1111111111000100	16	5	1	1
1111111111000101	16	6	1	1
1111111111000110	16	7	1	1
1111111111000111	16	8	1	1
1111111111001000	16	9	1	1
1111111111001001	16	10	1	1
1111111111001010	16	11	1	1
1111111111001011	16	12	1	1
1111111111001100	16	13	1	1
1111111111001101	16	14	1	1
1111111111001110	16	15	1	1
1111111111001111	16	16	1	1
1111111111010000	16	17	1	1
1111111111010001	16	18	1	1
1111111111010010	16	19	1	1
1111111111010011	16	20	1	1
1111111111010100	16	21	1	1
1111111111010101	16	22	1	1
1111111111010110	16	23	1	1
1111111111010111	16	24	1	1
1111111111011000	16	25	1	1
1111111111011001	16	26	1	1
1111111111011010	16	27	1	1
1111111111011011	16	28	1	1
1111111111011100	16	29	1	1
1111111111011101	16	30	1	1
1111111111011110	16	31	1	1
1111111111011111	16	32	1	1
1111111111100000	16	33	1	1
1111111111100001	16	34	1	1
1111111111100010	16	35	1	1
1111111111100011	16	36	1	1
1111111111100100	16	37	1	1
1111111111100101	16	38	1	1
1111111111100110	16	39	1	1
1111111111100111	16	40	1	1
1111111111101000	16	41	1	1
1111111111101001	16	42	1	1
1111111111101010	16	43	1	1

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
1111111111101011	16	44	1	1
1111111111101100	16	45	1	1
1111111111101101	16	46	1	1
1111111111101110	16	47	1	1
1111111111101111	16	48	1	1
1111111111110000	16	49	1	1
1111111111110001	16	50	1	1
1111111111110010	16	51	1	1
1111111111110011	16	52	1	1
1111111111110100	16	53	1	1
1111111111110101	16	54	1	1

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
1111111111110110	16	55	1	1
1111111111110111	16	56	1	1
1111111111111000	16	57	1	1
1111111111111001	16	58	1	1
1111111111111010	16	59	1	1
1111111111111011	16	60	1	1
1111111111111100	16	61	1	1
1111111111111101	16	62	1	1
1111111111111110	16	63	1	1
1111111111111111	16	64	1	1

[1] See Section 7.3.1.1 for the definition of the End-Of-Block (EOB) codeword

Table E.14 – Run VLC Table for Compression ID 1251

Codeword	Length	Run
0	1	1
100	3	2
101	3	3
1100	4	4
11010	5	5
11011	5	6
11100	5	7
111010	6	8
1110110	7	9
1110111	7	10
1111000	7	11
11110010	8	12
111100110	9	13
111100111	9	14
1111010000	10	15
1111010001	10	16
1111010010	10	17
1111010011	10	18
1111010100	10	19
1111010101	10	20
1111010110	10	21
1111010111	10	22
1111011000	10	23
1111011001	10	24
1111011010	10	25
1111011011	10	26
1111011100	10	27
1111011101	10	28
1111011110	10	29
1111011111	10	30
1111100000	10	31

Codeword	Length	Run
1111100001	10	32
1111100010	10	33
1111100011	10	34
1111100100	10	35
1111100101	10	36
1111100110	10	37
1111100111	10	38
1111101000	10	39
1111101001	10	40
1111101010	10	41
1111101011	10	42
1111101100	10	43
1111101101	10	44
1111101110	10	45
1111101111	10	46
1111110000	10	47
1111110001	10	48
1111110010	10	49
1111110011	10	50
1111110100	10	51
1111110101	10	52
1111110110	10	53
1111110111	10	54
1111111000	10	55
1111111001	10	56
1111111010	10	57
1111111011	10	58
1111111100	10	59
1111111101	10	60
1111111110	10	61
1111111111	10	62

Table E.15 – DC Differential VLC Table for Compression ID 1251

Codeword	η
000	0
1100	1
1101	2
001	3
010	4
011	5
100	6
101	7
1110	8
11110	9
111110	10
111111	11

VCL Tables for Compression ID 1252 and 1258

Table E.16 – Amplitude VLC Tables for Compression ID 1252 and 1258

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
00	2	1	0	0
01	2	1	1	0
100	3	2	0	0
1010	4	3	0	0
1011	4	2	1	0
1100	4	EOB[1]	0	0
11010	5	4	0	0
11011	5	5	0	0
111000	6	6	0	0
111001	6	7	0	0
111010	6	3	1	0
1110110	7	8	0	0
1110111	7	9	0	0
1111000	7	10	0	0
11110010	8	11	0	0
11110011	8	12	0	0
11110100	8	13	0	0
11110101	8	14	0	0
11110110	8	4	1	0
11110111	8	5	1	0
111110000	9	15	0	0
111110001	9	16	0	0
111110010	9	17	0	0
111110011	9	18	0	0
111110100	9	6	1	0
1111101010	10	19	0	0
1111101011	10	20	0	0
1111101100	10	21	0	0
1111101101	10	22	0	0
1111101110	10	23	0	0
1111101111	10	24	0	0
1111110000	10	7	1	0
1111110001	10	8	1	0
11111100100	11	25	0	0
11111100101	11	26	0	0
11111100110	11	27	0	0
11111100111	11	28	0	0
11111101000	11	29	0	0
11111101001	11	30	0	0
11111101010	11	31	0	0
11111101011	11	32	0	0
11111101100	11	9	1	0
11111101101	11	10	1	0
111111011100	12	33	0	0
111111011101	12	34	0	0
111111011110	12	35	0	0
111111011111	12	36	0	0
111111100000	12	37	0	0
111111100001	12	38	0	0
111111100010	12	39	0	0
111111100011	12	40	0	0
111111100100	12	41	0	0
111111100101	12	11	1	0
111111100110	12	12	1	0
111111100111	12	13	1	0
1111111010000	13	42	0	0

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
1111111010001	13	43	0	0
1111111010010	13	44	0	0
1111111010011	13	45	0	0
1111111010100	13	46	0	0
1111111010101	13	47	0	0
1111111010110	13	48	0	0
1111111010111	13	49	0	0
1111111011000	13	50	0	0
1111111011001	13	51	0	0
1111111011010	13	52	0	0
1111111011011	13	53	0	0
1111111011100	13	14	1	0
1111111011101	13	15	1	0
1111111011110	13	16	1	0
11111110111110	14	54	0	0
11111110111111	14	55	0	0
11111111000000	14	56	0	0
11111111000001	14	57	0	0
11111111000010	14	58	0	0
11111111000011	14	59	0	0
11111111000100	14	60	0	0
11111111000101	14	61	0	0
11111111000110	14	62	0	0
11111111000111	14	63	0	0
11111111001000	14	64	0	0
11111111001001	14	1	0	1
11111111001010	14	2	0	1
11111111001011	14	3	0	1
11111111001100	14	17	1	0
11111111001101	14	18	1	0
11111111001110	14	19	1	0
11111111001111	14	20	1	0
111111110100000	15	4	0	1
111111110100001	15	5	0	1
111111110100010	15	6	0	1
111111110100011	15	7	0	1
111111110100100	15	8	0	1
111111110100101	15	9	0	1
111111110100110	15	10	0	1
111111110100111	15	11	0	1
111111110101000	15	12	0	1
111111110101001	15	13	0	1
111111110101010	15	14	0	1
111111110101011	15	15	0	1
111111110101100	15	16	0	1
111111110101101	15	17	0	1
111111110101110	15	18	0	1
111111110101111	15	19	0	1
111111110110000	15	20	0	1
111111110110001	15	21	0	1
111111110110010	15	21	1	0
111111110110011	15	22	1	0
111111110110100	15	23	1	0
111111110110101	15	24	1	0
111111110110110	15	25	1	0
1111111101101110	16	22	0	1

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
1111111101101111	16	23	0	1
1111111101110000	16	24	0	1
1111111101110001	16	25	0	1
1111111101110010	16	26	0	1
1111111101110011	16	27	0	1
1111111101110100	16	28	0	1
1111111101110101	16	29	0	1
1111111101110110	16	30	0	1
1111111101110111	16	31	0	1
1111111101111000	16	32	0	1
1111111101111001	16	33	0	1
1111111101111010	16	34	0	1
1111111101111011	16	35	0	1
1111111101111100	16	36	0	1
1111111101111101	16	37	0	1
1111111101111110	16	38	0	1
1111111101111111	16	39	0	1
1111111110000000	16	40	0	1
1111111110000001	16	41	0	1
1111111110000010	16	42	0	1
1111111110000011	16	43	0	1
1111111110000100	16	44	0	1
1111111110000101	16	45	0	1
1111111110000110	16	46	0	1
1111111110000111	16	47	0	1
1111111110001000	16	48	0	1
1111111110001001	16	49	0	1
1111111110001010	16	50	0	1
1111111110001011	16	51	0	1
1111111110001100	16	52	0	1
1111111110001101	16	53	0	1
1111111110001110	16	54	0	1
1111111110001111	16	55	0	1
1111111110010000	16	56	0	1
1111111110010001	16	57	0	1
1111111110010010	16	58	0	1
1111111110010011	16	59	0	1
1111111110010100	16	60	0	1
1111111110010101	16	61	0	1
1111111110010110	16	62	0	1
1111111110010111	16	63	0	1
1111111110011000	16	64	0	1
1111111110011001	16	26	1	0
1111111110011010	16	27	1	0
1111111110011011	16	28	1	0
1111111110011100	16	29	1	0
1111111110011101	16	30	1	0
1111111110011110	16	31	1	0
1111111110011111	16	32	1	0
1111111110100000	16	33	1	0
1111111110100001	16	34	1	0
1111111110100010	16	35	1	0
1111111110100011	16	36	1	0
1111111110100100	16	37	1	0
1111111110100101	16	38	1	0
1111111110100110	16	39	1	0
1111111110100111	16	40	1	0
1111111110101000	16	41	1	0
1111111110101001	16	42	1	0
1111111110101010	16	43	1	0
1111111110101011	16	44	1	0
1111111110101100	16	45	1	0

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
1111111110101101	16	46	1	0
1111111110101110	16	47	1	0
1111111110101111	16	48	1	0
1111111110110000	16	49	1	0
1111111110110001	16	50	1	0
1111111110110010	16	51	1	0
1111111110110011	16	52	1	0
1111111110110100	16	53	1	0
1111111110110101	16	54	1	0
1111111110110110	16	55	1	0
1111111110110111	16	56	1	0
1111111110111000	16	57	1	0
1111111110111001	16	58	1	0
1111111110111010	16	59	1	0
1111111110111011	16	60	1	0
1111111110111100	16	61	1	0
1111111110111101	16	62	1	0
1111111110111110	16	63	1	0
1111111110111111	16	64	1	0
1111111111000000	16	1	1	1
1111111111000001	16	2	1	1
1111111111000010	16	3	1	1
1111111111000011	16	4	1	1
1111111111000100	16	5	1	1
1111111111000101	16	6	1	1
1111111111000110	16	7	1	1
1111111111000111	16	8	1	1
1111111111001000	16	9	1	1
1111111111001001	16	10	1	1
1111111111001010	16	11	1	1
1111111111001011	16	12	1	1
1111111111001100	16	13	1	1
1111111111001101	16	14	1	1
1111111111001110	16	15	1	1
1111111111001111	16	16	1	1
1111111111010000	16	17	1	1
1111111111010001	16	18	1	1
1111111111010010	16	19	1	1
1111111111010011	16	20	1	1
1111111111010100	16	21	1	1
1111111111010101	16	22	1	1
1111111111010110	16	23	1	1
1111111111010111	16	24	1	1
1111111111011000	16	25	1	1
1111111111011001	16	26	1	1
1111111111011010	16	27	1	1
1111111111011011	16	28	1	1
1111111111011100	16	29	1	1
1111111111011101	16	30	1	1
1111111111011110	16	31	1	1
1111111111011111	16	32	1	1
1111111111100000	16	33	1	1
1111111111100001	16	34	1	1
1111111111100010	16	35	1	1
1111111111100011	16	36	1	1
1111111111100100	16	37	1	1
1111111111100101	16	38	1	1
1111111111100110	16	39	1	1
1111111111100111	16	40	1	1
1111111111101000	16	41	1	1
1111111111101001	16	42	1	1
1111111111101010	16	43	1	1

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
1111111111101011	16	44	1	1
1111111111101100	16	45	1	1
1111111111101101	16	46	1	1
1111111111101110	16	47	1	1
1111111111101111	16	48	1	1
1111111111110000	16	49	1	1
1111111111110001	16	50	1	1
1111111111110010	16	51	1	1
1111111111110011	16	52	1	1
1111111111110100	16	53	1	1
1111111111110101	16	54	1	1

Codeword	Length	Amp	Run Flag F_{run}	Index Flag F_{index}
1111111111110110	16	55	1	1
1111111111110111	16	56	1	1
1111111111111000	16	57	1	1
1111111111111001	16	58	1	1
1111111111111010	16	59	1	1
1111111111111011	16	60	1	1
1111111111111100	16	61	1	1
1111111111111101	16	62	1	1
1111111111111110	16	63	1	1
1111111111111111	16	64	1	1

[1] See Section 7.3.1.1 for the definition of the End-Of-Block (EOB) codeword

Table E.17 – Run VLC Table for Compression ID 1252 and 1258

Codeword	Length	Run
0	1	1
100	3	2
101	3	3
1100	4	4
11010	5	5
11011	5	6
11100	5	7
111010	6	8
1110110	7	9
1110111	7	10
1111000	7	11
11110010	8	12
111100110	9	13
111100111	9	14
1111010000	10	15
1111010001	10	16
1111010010	10	17
1111010011	10	18
1111010100	10	19
1111010101	10	20
1111010110	10	21
1111010111	10	22
1111011000	10	23
1111011001	10	24
1111011010	10	25
1111011011	10	26
1111011100	10	27
1111011101	10	28
1111011110	10	29
1111011111	10	30
1111100000	10	31

Codeword	Length	Run
1111100001	10	32
1111100010	10	33
1111100011	10	34
1111100100	10	35
1111100101	10	36
1111100110	10	37
1111100111	10	38
1111101000	10	39
1111101001	10	40
1111101010	10	41
1111101011	10	42
1111101100	10	43
1111101101	10	44
1111101110	10	45
1111101111	10	46
1111110000	10	47
1111110001	10	48
1111110010	10	49
1111110011	10	50
1111110100	10	51
1111110101	10	52
1111110110	10	53
1111110111	10	54
1111111000	10	55
1111111001	10	56
1111111010	10	57
1111111011	10	58
1111111100	10	59
1111111101	10	60
1111111110	10	61
1111111111	10	62

Table E.18 – DC Differential VLC Table for Compression ID 1252 and 1258

Codeword	η
000	0
1100	1
1101	2
001	3
010	4
011	5
100	6
101	7
1110	8
11110	9
111110	10
111111	11

Annex F System Overview (Informative)

This informative annex provides the context of how the VC-3 bitstream and decoding process relate to other system level aspects. Figure F.1 illustrates the scope of this standard and where the VC-3 decoding process lies within a typical application. SMPTE ST 2019-3 describes the mapping of CBR VC-3 and AES3 data over SDTI.

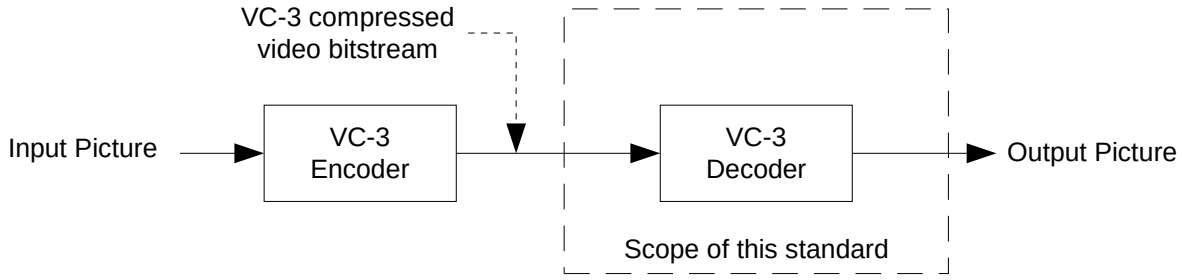

Figure F.1 – Scope of the VC-3 compression standard

The encoding process is not defined in this standard; however, Figure F.2 illustrates a typical encoder data flow. The process shown is not the only possible compression method but is representative of a typical encoding approach. The main elements of a VC-3 video encoder are:

- The Forward Discrete Cosine Transform which converts an input picture to a DCT-based, spatial frequency representation.
- The Zig-zag element which re-orders the DCT block transform elements into an order that is efficient for entropy encoding.
- The Rate-control element which ensures that the desired compression rate is achieved by controlling the quantization values applied to the DCT coefficients.
- The Quantization element which quantizes the DCT coefficients to reduce bandwidth required to represent the input picture.
- The Entropy Encoding element which losslessly encodes the quantized DCT coefficients.
- The Bitstream Formation element which formats the entropy codewords into a valid VC-3 compressed bitstream.

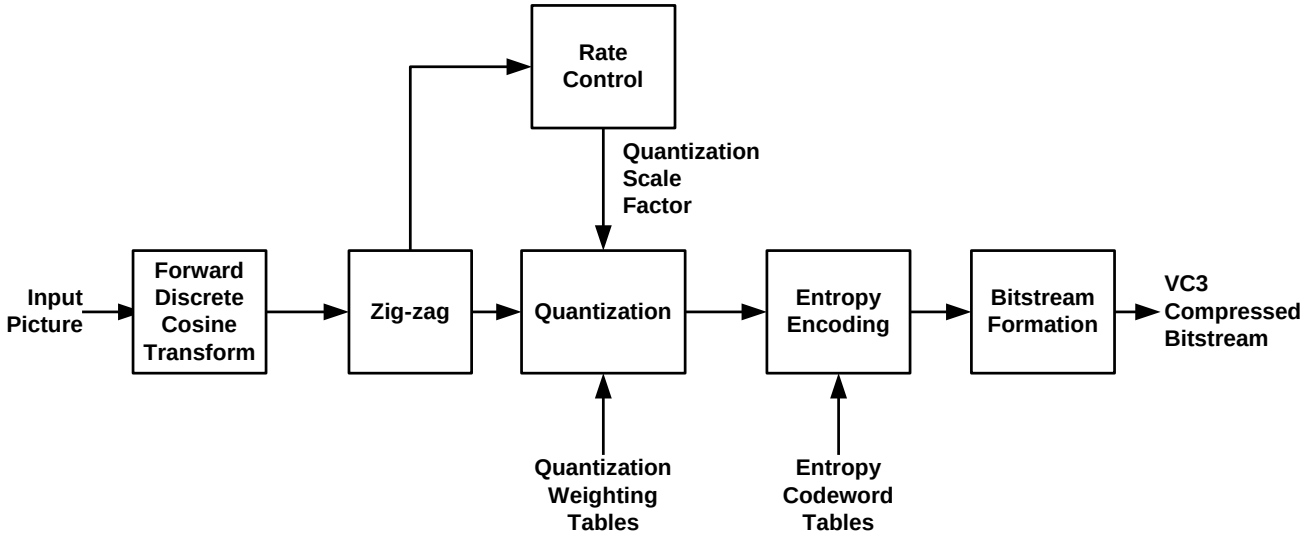

Figure F.2 – Overview of the compression process (Video and DCT-based Alpha)

The main elements of the lossless Alpha encoding process are

- The Alpha Zig-Zag re-orders Alpha lines to avoid continuity jumps during differential encoding.
- The Difference Prediction determines which prediction type results in the minimal difference for the re-ordered sequence.
- The RLE encoding performs run-length encoding of the prediction/difference pairs.
- The Size Control compares the result to the size achieved for un-encoded binary storage, selecting the more effective encoding method.
- Bitstream Formation formats the selected encoding into a valid VC-3 Alpha bitstream.

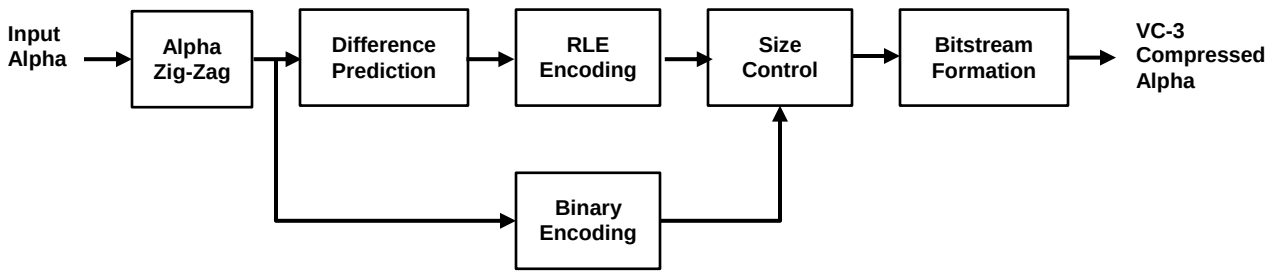

Figure F.3 – Overview of the lossless Alpha compression process

SMPTE RP 2019-2 provides the details regarding VC-3 bitstream and decoder conformance testing.

Annex G Compressed Bitrates (Informative)

Table G.1 below provides compressed bitrates by Compression ID (HD) for some typical video frame rates. The bitrate is rounded to the nearest 5 Mbps.

Table G.1 – Compressed bitrates for HD rasters rounded to the nearest 5 Mbps

					Frame rate (fps)				
					23.98	25	29.97	50	59.94
Compression ID	Source scan type	Samples per line	Active lines per frame	Compressed bytes per frame	Mbps	Mbps	Mbps	Mbps	Mbps
1235	progressive	1920	1080	917504	180	185	220	370	440
1237	progressive			606208	120	120	145	240	290
1238	progressive			917504	180	185	220	370	440
1241	interlaced			917504	180	185	220	370	440
1242	interlaced			606208	120	120	145	240	290
1243	interlaced			917504	180	185	220	370	440
1244	interlaced	1440	1080	606208	120	120	145	240	290
1250	progressive	1280	720	458752	90	90	110	185	220
1251	progressive			458752	90	90	110	185	220
1252	progressive			303104	60	60	70	120	145
1253	progressive	1920	1080	188416	35	40	45	75	90
1256	progressive	1920	1080	1835008	350	370	440	735	880
1258	progressive	960	720	212992	40	40	50	85	100
1259	progressive	1440	1080	417792	80	85	100	170	200
1260	interlaced	1440	1080	417792	80	85	100	170	200

Table G.2 below provides examples of compressed bitrates by Compression ID for some typical video frame rates and selected RI raster sizes. The bitrate is rounded to the nearest 5 Mbps.

Table G.2 – Compressed bitrates (CBR, no alpha) for typical RI rasters rounded to the nearest 5 Mbps

			Frame rate family[1] Bitrate in Mbps		
Compression ID	Format [width x height]	Compressed bytes per frame	23.98/ 24 [2]	25	29.97/ 30 [2]
1270 (444, 4:4:4/RGB, 10/12 bit)	**16:9 SD** (1/4 FHD, 1/16 UHD-1) [960 x 540]	458752	90	90	110
	FHD (1/4 UHD-1) [1920 x 1080]	1835008	350	370	440
	DCI 2K [2048 x 1080]	1957888	375	390	470
	Full aperture 35mm film scan [2048 x 1556]	2820875	540	565	680
	UHD-1 (QFHD) [3840 x 2160]	7286061	1400	1460	1750
	DCI 4K [4096 x 2160]	7771799	1490	1555	1865
1271 (HQX, 4:2:2/4:2:0, 10/12 bit)	**16:9 SD** (1/4 FHD, 1/16 UHD-1) [960 x 540]	229376	45	45	55
	FHD (1/4 UHD-1) [1920 x 1080]	917504	180	180	220
	DCI 2K [2048 x 1080]	978671	190	200	235
	Full aperture 35mm film scan [2048 x 1556]	1410438	270	280	340
	UHD-1 (QFHD) [3840 x 2160]	3643031	700	730	875
	DCI 4K [4096 x 2160]	3885899	745	780	935
1272 (HQ, 4:2:2/4:2:0, 8 bit)	**16:9 SD** (1/4 FHD, 1/16 UHD-1) [960 x 540]	229376	45	45	55
	FHD (1/4 UHD-1) [1920 x 1080]	917504	180	180	220
	DCI 2K [2048 x 1080]	978671	190	200	235
	Full aperture 35mm film scan [2048 x 1556]	1410438	270	280	340
	QFHD (UHD-1) [3840 x 2160]	3643031	700	730	875
	DCI 4K [4096 x 2160]	3885899	745	780	935

[1] A frame rate family is formed by the base frame rate and its power of 2 multiples. For example the 25 fps frame rate family contains the frequencies { 25 (base), 50, 100, 200, …} fps. The bitrates for these multiple are obtained by multiplying the bitrate of the base frequency with the respective power of 2 (*1, *2, *4, *8, …).

[2] The 0.1% difference between these frame rates is not significant as the bitrates are rounded to multiples of 5 Mbps.

Compression ID	Format	Compressed bytes per frame	Frame rate family[1] Bitrate in Mbps		
			23.98/ 24 [2]	25	29.97/ 30 [2]
1273 (SQ, 4:2:2/4:2:0, 8 bit)	**16:9 SD** (1/4 FHD, 1/16 UHD-1) [960 x 540]	151552	30	30	35
	FHD (1/4 UHD-1) [1920 x 1080]	606208	115	120	145
	DCI 2K [2048 x 1080]	646622	125	130	155
	Full aperture 35mm film scan [2048 x 1556]	931896	180	185	225
	QFHD (UHD-1) [3840 x 2160]	2407002	460	480	580
	DCI 4K [4096 x 2160]	2567469	490	510	615
1274 (LB, 4:2:2/4:2:0, 8 bit)	**16:9 SD** (1/4 FHD, 1/16 UHD-1) [960 x 540]	47104	10	10	10
	FHD (1/4 UHD-1) [1920 x 1080]	188416	35	40	45
	DCI 2K [2048 x 1080]	200977	40	40	50
	Full aperture 35mm film scan [2048 x 1556]	289643	60	60	70
	QFHD (UHD-1) [3840 x 2160]	748122	145	150	180
	DCI 4K [4096 x 2160]	797997	155	160	190

[1] A frame rate family is formed by the base frame rate and its power of 2 multiples. For example the 25 fps frame rate family contains the frequencies { 25 (base), 50, 100, 200, …} fps. The bitrates for these multiple are obtained by multiplying the bitrate of the base frequency with the respective power of 2 (*1, *2, *4, *8, …).

[2] The 0.1% difference between these frame rates is not significant as the bitrates are rounded to multiples of 5 Mbps.

Annex H VC-3 Profiles and Levels (Informative)

Table H.1 defines a reference scheme for VC-3 parameter combinations, which shows preferred values for displaying VC-3 categories to the user. For example, media using Compression ID 1271 shows preferred values to be presented using the qualifiers “RI HQX”.

Alpha support has been omitted in the RI levels for clarity.

Table H.1 – VC-3 Levels and Profiles

Profile

		HD						RI					
Level	**Name**	**CID**	**Scan Type**	**Bit depth**	**Chroma sub-sampling**	**CR[5]**	**Mbps[1] (@29.97)**	**CID**	**Scan Type**	**Bit depth**	**Chroma sub-samping**	**CR[5]**	**Mbps[2] (DCI 4K @ 59.94)**
444	4:4:4	1256	P	10	RGB	4.2	420	1270	P	12/10	RGB 4:4:4	5.0/4.2 5.0/4.2	3730
HQX	High Quality, extended	1235 1241 1250	P I P		4:2:2	5.7 5.7 5.0	210	1271	P		4:2:2 4:2:0	6.8/5.7 5.1/4.2	1865
HQ	High Quality	1238 1243 1251	P I P	8		4.5 4.5 4.0	210	1272	P	8		4.5 3.9	1865
SQ	Standard Quality	1237 1242 1252	P I P			6.9 6.9 6.0	140 140 70	1273	P			6.9 5.1	1230
LB	Low Bandwidth	1253	P			22.0	50	1274	P			22.0 16.5	385
TR	Thin Raster	*1244*[3] 1258 1259 1260	I P P I			*5.1* 6.5 7.4 7.4	*140*[3] 50 100 100	—[4]					

[1] Only the bitrate at 29.97 fps for the resolution defined by the CID is shown here to simplify identification of the comparison of the level differences.

[2] The bitrate shown for the RI profiles is calculated at 59.94 fps for DCI 4K, CBR, no alpha. For FullHD@29.97 the values match their HD counterparts at the same level.

[3] The operating points for Compression ID 1244 are retained (in this nomenclature) for backwards compatibility only, providing a higher data rate compared to Compression ID 1260, to which it is otherwise identical.

[4] Thin Raster, as used in the HD profile, is not supported in the RI profile. RI can compress arbitrary resolutions (including Thin Raster resolutions).

[5] The compression ratio CR (uncompressed/compressed size) uses the ideal bitstream size (closest possible packing) of the uncompressed signal at the specified bit depth and chroma sub-sampling for comparisons against the compressed size. For example a 1920 x 1080, 4:2:0, 12-bits frame has an “ideal” bit stream size of 1920*1080*(6/4)*12 = 37,324,800 bits.

Annex I Bibliography (Informative)

SMPTE ST 291-1:2011, Ancillary Data Packet and Space Formatting

SMPTE RP 2019-2:2016, VC-3 Decoder and Bitstream Conformance

SMPTE ST 2019-3:2008, VC-3 Type Data Stream and Mapping over SDTI

SMPTE ST 2019-4:2016, Mapping VC-3 Coding Units into the MXF Generic Container

Annex J Revision History

This annex provides an overview of the essential changes introduced with each revision of this standard.

SMPTE ST 2019-1:2008: Initial specification

SMPTE ST 2019-1:2014:

- Add 1 RGB compression type
- Add 3 Thin-Raster compression types

SMPTE ST 2019-1:2016:

- Remove raster resolution constraints
- Add 4:4:4 and 4:2:0 YC_BC_R compression types
- Add 12-bit support
- Add ITU Rec BT.2020 and optional out-of-band encodings
- Add Alpha channel, DCT-based and loss-less
- Add VBR encoding