INTERNATIONAL STANDARD

ISO/IEC 18181-1

First edition
2022-03

Information technology — JPEG XL image coding system —

Part 1: Core coding system

Technologies de l'information — Système de codage d'images JPEG XL —

Partie 1: Système de codage de noyau

Reference number
ISO/IEC 18181-1:2022(E)

Contents

Page

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives or www.iec.ch/members_experts/refdocs).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents) or the IEC list of patent declarations received (see https://patents.iec.ch).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see www.iso.org/iso/foreword.html. In the IEC, see www.iec.ch/understanding-standards.

This document was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information*.

A list of all parts in the ISO/IEC 18181 series can be found on the ISO and IEC websites.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html and www.iec.ch/national-committees.

Introduction

The International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC) draw attention to the fact that it is claimed that compliance with this document may involve the use of a patent.

ISO and IEC take no position concerning the evidence, validity and scope of this patent right.

The holder of this patent right has assured ISO and IEC that they are willing to negotiate licences under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statement of the holder of this patent right is registered with ISO and IEC. Information may be obtained from the patent database available at www.iso.org/patents.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights other than those in the patent database. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

Information technology — JPEG XL image coding system —

Part 1:
Core coding system

1 Scope

This document defines a set of compression methods for coding one or more images of bi-level, continuous-tone greyscale, or continuous-tone colour, or multichannel digital samples.

This document:

— specifies decoding processes for converting compressed image data to reconstructed image data;

— specifies a codestream syntax containing information for interpreting the compressed image data;

— provides guidance on encoding processes for converting source image data to compressed image data.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

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

ISO/IEC 60559, *Information technology — Microprocessor Systems — Floating-Point arithmetic*

IEC 61966-2-1, *Multimedia systems and equipment — Colour measurement and management — Part 2-1: Colour management — Default RGB colour space — sRGB*

Rec. ITU-R BT.2100-2, *Image parameter values for high dynamic range television for use in production and international programme exchange*

Rec. ITU-R BT.709-6, *Parameter values for the HDTV standards for production and international programme exchange*

SMPTE ST 428-1, *D-Cinema distribution master — Image characteristics*

3 Terms and definitions

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

ISO and IEC maintain terminology databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp

— IEC Electropedia: available at https://www.electropedia.org/

3.1 Data storage

3.1.1
byte
8 consecutive bits encoding a value between 0 and 255

3.1.2
big endian
value representation with bytes in most to least-significant order

3.1.3
bitstream
sequence of bytes from which bits are read starting from the least-significant bit of the first byte

3.1.4
codestream
bitstream representing compressed image data

3.1.5
bundle
structured data consisting of one or more fields

3.1.6
field
numerical value or bundle, or an array of either

3.1.7
histogram
array of unsigned integers representing a probability distribution, used for entropy coding

3.1.8
set
unordered collection of elements

3.2 Inputs

3.2.1
pixel
vector of dimension corresponding to the number of channels, consisting of samples

3.2.2
sample
integer or real value, of which there is one per channel per pixel

3.2.3
greyscale
image representation in which each pixel is defined by a single sample representing intensity (either luminance or luma depending on the ICC profile)

3.2.4
continuous-tone image
image having samples consisting of more than one bit

3.2.5
opsin
photosensitive pigments in the human retina, having dynamics approximated by the XYB colour space

3.2.6
burst
sequences of images typically captured with identical settings

3.2.7
animation
series of pictures and timing delays to display as a video medium

3.2.8
composite
series of images that are superimposed

3.2.9
frame
single image (possibly part of a burst or animation or composite)

3.2.10
preview
lower-fidelity rendition of one of the frames (e.g. lower resolution), or a frame that represents the entire content of all frames

3.3 Processes

3.3.1
decoding process
process which takes as its input a codestream and outputs a continuous-tone image

3.3.2
decoder
embodiment of a decoding process

3.3.3
encoding process
process which takes as its input continuous-tone image(s) and outputs compressed image data in the form of a codestream

3.3.4
encoder
embodiment of an encoding process

3.3.5
lossless
descriptive term for encoding and decoding processes in which the output of a decoding procedure is identical to the input to the encoding procedure

3.3.6
lossy
descriptive term for encoding and decoding processes which are not lossless

3.3.7
upsampling
procedure by which the (nominal) spatial resolution of a channel is increased

3.3.8
downsampling
procedure by which the spatial resolution of a channel is reduced

3.3.9
entropy encoding
lossless procedure designed to convert a sequence of input symbols into a sequence of bits such that the average number of bits per symbol approaches the entropy of the input symbols

3.3.10
entropy encoder
embodiment of an entropy encoding procedure

3.3.11
entropy decoding
lossless procedure which recovers the sequence of symbols from the sequence of bits produced by the entropy encoder

3.3.12
entropy decoder
embodiment of an entropy decoding procedure

3.3.13
Gabor-like transform
convolution with default or signalled 3x3 kernel for deblocking

3.3.14
tick
unit of time such that animation frame durations are integer multiples of the tick duration

3.4 Image organization

3.4.1
grid
2-dimensional array; `a [x, y]` means addressing an element of grid `a` at row `y` and column `x`. Where so specified, addressing elements with coordinates outside of bounding rectangle (`x < 0`, or `y < 0`, or `x >= width`, or `y >= height`) is allowed

3.4.2
sample grid
common coordinate system for all samples of an image, with top-left coordinates (0, 0), the first coordinate increasing towards the right, and the second increasing towards the bottom

3.4.3
channel
component
rectangular array of samples having the same designation, regularly aligned along a sample grid

3.4.4
rectangle
rectangular area within a channel or grid

3.4.5
width
width in samples of a sample grid or a rectangle

3.4.6
height
height in samples of a sample grid or a rectangle

3.4.7
raster order
access pattern from left to right in the top row, then in the row below and so on

3.4.8
naturally aligned
positioning of a power-of-two sized rectangle such that its top and left coordinates are divisible by its width and height, respectively

3.4.9
block
naturally aligned square rectangle covering up to 8 × 8 input pixels

3.4.10
group
naturally aligned square rectangle covering up to $2^n \times 2^n$ (with n between 7 and 10, inclusive) input pixels

3.4.11
table of contents
data structure that enables seeking to a group or the next frame within a codestream

3.4.12
section
part of the codestream with an offset and length that are stored in a frame's table of contents

3.5 DCT

3.5.1
coefficient
input value to the inverse DCT

3.5.2
quantization
method of reducing the precision of individual coefficients

3.5.3
varblock
variable-size rectangle of input pixels

3.5.4
dct_block
an array with 64 elements corresponding to DCT coefficients of a (8 × 8) block

3.5.5
var-DCT
lossy encoding of a frame that applies DCT to varblocks

3.5.6
LF coefficient
lowest frequency DCT coefficient, containing the average value of a block or the lowest-frequency coefficient within the 8 × 8 rectangle of a varblock of size greater than 8 × 8

3.5.7
HF coefficients
all DCT coefficients apart from the LF coefficients, i.e. the high frequency coefficients

3.5.8
pass
data enabling decoding of successively higher resolutions

3.5.9
LF group
$2^n \times 2^n$ LF values from a naturally aligned rectangle covering up to $2^{n+3} \times 2^{n+3}$ input pixels

3.5.10
quantization weight
factor that a quantized coefficient is multiplied by prior to application of the inverse DCT in the decoding process

3.5.11
channel decorrelation
method of reducing total encoded entropy by removing correlations between channels

3.5.12
channel correlation factor
factor by which a channel should be multiplied by before adding it to another channel to undo the channel decorrelation process

4 Abbreviated terms

DCT: discrete cosine transform (DCT-II as specified in I.2)

IDCT: inverse discrete cosine transform (DCT-III as specified in I.2)

LF: N / 8 × M / 8 square of lowest frequency coefficients of N × M DCT coefficients

RGB: additive colour model with red, green, blue channels

LMS: absolute colour space representing the response of cone cells in the human eye

XYB: absolute colour space based on gamma-corrected LMS, in which X is derived from the difference between L and M, Y is an average of L and M (behaves similarly to luminance), and B is derived from the S ("blue") channel

5 Conventions

5.1 Mathematical symbols

`[a, b], (c, d), [e, f)`	closed or open or half-open intervals containing all integers or real numbers x (depending on context) such that `a ≤ x ≤ b, c < x < d, e ≤ x < f.`
`{a, b, c}`	ordered sequence of elements
`π`	the smallest positive zero of the sine function

5.2 Functions

`sqrt(x)`	square root, such that `(sqrt(x))`^{2} `== x` and `sqrt(x) >= 0`. Undefined for x < 0.
`cbrt(x)`	cube root, such that `(cbrt(x))`^{3} `== x`.
`cos(r)`	cosine of the angle r (in radians)
`erf(x)`	Gauss error function: `erf(x)` $= \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$
`log(x)`	natural logarithm of x. Undefined for x <= 0.
`log2(x)`	base-two logarithm of x. Undefined for x <= 0.
`floor(x)`	the largest integer that is less than or equal to x
`ceil(x)`	the smallest integer that is greater than or equal to x
`abs(x)`	absolute value of x: equal to -x if x < 0, otherwise x
`sign(x)`	sign of x, 0 if x is 0, +1 if x is positive, -1 if x is negative
`UnpackSigned(u)`	equivalent to u / 2 if u is even, and -(u + 1) / 2 if u is odd

`clamp(x, lo, hi)`	equivalent to `min({max({lo, x}), hi})`
`InterpretAsF16(u)`	the real number resulting from interpreting the unsigned 16-bit integer `u` as a binary16 floating-point number representation (cf. ISO/IEC 60559)
`InterpretAsF32(u)`	the real number resulting from interpreting the unsigned 32-bit integer `u` as a binary32 floating-point number representation (cf. ISO/IEC 60559)
`len(a)`	length (number of elements) of array a
`sum(a)`	sum of all elements of the array/tuple/sequence a
`max(a)`	maximal element of the array/tuple/sequence a
`min(a)`	smallest element of the array/tuple/sequence a

5.3 Operators

This document uses the operators defined by the C++ programming language [2], with the following differences:

`×`	multiplication
`×=`	`a ×= b` is equivalent to `a = a × b`
`/`	division of real numbers without truncation or rounding. Division by zero is undefined.
x^y	exponentiation, x to the power of y
`<<`	left shift: `x << s` is defined as $x \times 2^s$
`>>`	right shift: `x >> s` is defined as `floor(x / 2`^{s}`)`
`Umod`	a `Umod` d is the unique integer r in [0, d) for which `a == r + q×d` for a suitable integer q
`Idiv`	a `Idiv` b is equivalent to `a / b`, rounded towards zero to an integer value

The order of precedence for these operators is listed below in descending order. 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).

Operators	**Type of operation**	**Associativity**
`++x, --x`	prefix increment/decrement	right to left
x^y	exponentiation	right to left
`!, ~`	logical/bitwise NOT	right to left
`×, /, Idiv, Umod`	multiplication, division, integer division, remainder	left to right
`+, -`	addition and subtraction	left to right
`<<, >>`	left shift and right shift	left to right
`< , >, <=, >=`	relational	left to right
`=`	assignment	right to left
`+=, -=, ×=`	compound assignment	right to left

5.4 Pseudocode

This document describes functionality using pseudocode formatted as follows:

```
// Informative comment
var = u(8);  // Defined in 9.2.1
if (var == 1) return;  // Stop executing this code snippet
[[Normative specification: var != 0]]
(out1, out2) = Function(var, kConstant);
```

Variables such as `var` are typically referenced by text outside the source code.

The semantics of this pseudocode are those of the C++ programming language [2], with the following exceptions:

— Symbols from 5.1 and functions from 5.2 are allowed;

— Multiplication, division, remainder and exponentiation are expressed as specified in 5.3;

— Functions can return tuples which unpack to variables as in the above example;

— `[[ ]]` enclose normative directives specified using prose;

— All integers are stored using two's complement;

— Expressions and variables of which types are omitted, are understood as real numbers.

Where unsigned integer wraparound and truncated division are required, `Umod` and `Idiv` (see 5.3) are used for those purposes.

Numbers with a 0x prefix are in base 16 (hexadecimal), and apostrophe (') characters inside them are understood to have no effect.

EXAMPLE 0x0001'0000 == 65536.

6 Functional concepts

6.1 Image organization

A channel is defined as a rectangular array of (integer or real) samples regularly aligned along a sample grid of `width` sample positions horizontally and `height` sample positions vertically. The number of channels may be 1 to 4099 (see `num_extra_channels` in A.6).

A pixel is defined as a vector of dimension corresponding to the number of channels, consisting of samples with a position matching that of the pixel. The index of a sample is numbered from 0 to number of channels - 1.

An image is defined as the two-dimensional array of pixels, and its width is `width` and height is `height`. Unless otherwise mentioned, channels are accessed in the following "raster order": left to right column within the topmost row, then left to right column within the row below the top, and so on until the rightmost column of the bottom row.

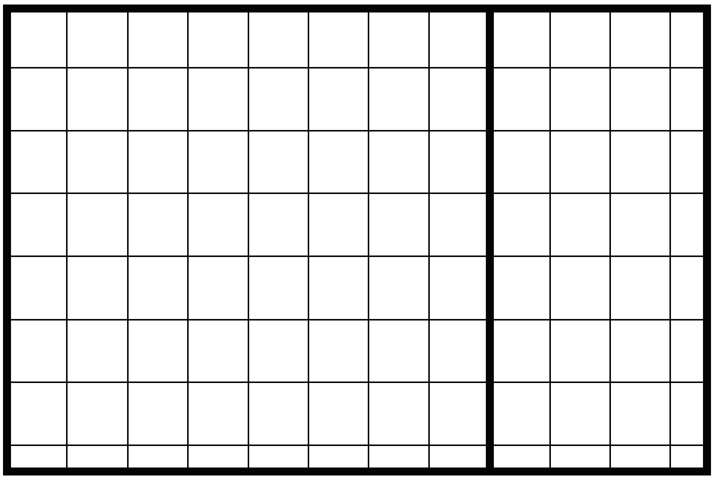
6.2 Group splitting

Channels are logically partitioned into naturally-aligned groups of `kGroupDim` × `kGroupDim` samples. The effective dimension of a group (i.e. how many pixels to read) can be smaller than `kGroupDim` for groups on the right or bottom of the image. The decoder ensures the decoded image has the dimensions specified in `SizeHeader` by cropping at the right and bottom as necessary. Unless otherwise specified, `kGroupDim` is 256.

LF groups likewise consist of `kGroupDim` × `kGroupDim` LF samples, with the possibility of a smaller effective size on the right and bottom of the image.

Groups can be decoded independently. A 'table of contents' stores the size (in bytes) of each group to allow seeking to any group. An optional permutation allows groups to be arranged in arbitrary order within the codestream.

EXAMPLE Figure 1 shows an example of the HF groups and LF groups of an image.



Frame: 2970×1868 pixels

HF groups:
11×7 groups of 256×256 pixels,
1×7 groups of 154×256 pixels,
7×1 groups of 256×76 pixels,
1 group of 154×76 pixels

LF groups:
1 group of 256×233 LF coefficients
(covering 2048×1868 pixels),
1 group of 116×233 LF coefficients
(covering 922×1868 pixels)

Figure 1 — Group splitting example

6.3 Codestream and bitstream

A bitstream is a finite sequence of bytes. A codestream is a bitstream that represent compressed image data and metadata. N bytes can also be viewed as 8 × N bits. The first 8 bits are the bits constituting the first byte, in least to most significant order, the next eight bits (again in least to most significant order) constitute the second byte, and so on. Unless otherwise specified, bits are read from the codestream as specified in 9.2.1.

NOTE Ordering bits from least to most significant allows using special CPU instructions to isolate the least-significant bits.

Subsequent Annexes or subclauses indicate some elements of the codestream are byte-aligned. For such elements, the decoder takes actions before and after reading the element as follows. Immediately before encountering the element, the decoder invokes ZeroPadToByte() (9.2.9). After finishing reading the element, the decoder invokes ZeroPadToByte() (9.2.9).