INTERNATIONAL STANDARD

ISO/IEC 21122-4

Second edition
2022-10

Information technology — JPEG XS low-latency lightweight image coding system —

Part 4: **Conformance testing**

Technologies de l'information — Système de codage d'images léger à faible latence JPEG XS —

Partie 4: Essais de conformité

Reference number
ISO/IEC 21122-4:2022(E)

© ISO/IEC 2022

COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2022

All rights reserved. Unless otherwise specified, or required in the context of its implementation, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents

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*.

This second edition cancels and replaces the first edition (ISO/IEC 21122-4:2020), which has been technically revised.

The main changes are as follows:

— reference test streams have been revised

— reference test streams for testing colour filter array (CFA) image compression have been added;

— reference test streams testing lossless coding have been added;

— reference test streams testing 4:2:0 sampled images has been added;

— a relaxed conformance point based on PSNR bounds was introduced.

A list of all parts in the ISO/IEC 21122 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.

© ISO/IEC 2022 – All rights reserved

Information technology — JPEG XS low-latency lightweight image coding system —

Part 4: **Conformance testing**

1 Scope

This document specifies the framework, concepts, methodology for testing, and criteria to be achieved to claim conformance to multiple parts of the ISO/IEC 21122 series. It lists the conformance testing procedures.

This document specifies:

— Conformance testing procedures for decoders implementing ISO/IEC 21122-1.

— Tests to check which conformance point an ISO/IEC 21122-1 decoder conforms to, that is, whether a decoder satisfies the error bounds required for strict or relaxed conformance.

— Conformance testing procedures for decoders implementing ISO/IEC 21122-3.

— Tests to check codestreams for conformance to ISO/IEC 21122-1. As such, it provides means to test whether encoder implementations generate syntactically correct codestreams, and whether codestreams generated by such implementations follow the requirements of a particular profile, level and sublevel, and the buffer model implied by them.

— Tests to check files for conformance to ISO/IEC 21122-3.

— Conformance testing procedures that allow testing whether codestreams conform to any of the profiles specified in ISO/IEC 21122-2.

— Conformance testing procedures that allow testing whether codestreams conform to the buffer model specified in ISO/IEC 21122-2 as part of a profile, level and sublevel.

— Codestreams, decoded images, and error metrics to be used within the decoder testing procedures.

— A buffer model test.

— Abstract test suites.

NOTE This document does not specify:

— Testing the reconstruction of a full resolution image from a subsampled image format. In particular, upsampling from 4:2:2 or 4:2:0 to 4:4:4 sampling is a non-normative extension and as such its testing is beyond the scope of this document.

— Testing the conversion of the sample values reconstructed by an ISO/IEC 21122-3 decoder to the target colour space by means of the colour specification box of ISO/IEC 21122-3.

— Testing of the composition of background and foreground for images reconstructed from ISO/IEC 21122-3 files or codestreams that contain auxiliary channels carrying opacity information.

— Testing of the interpolation of a colour filter array image to a full scale colour image; this process is not normatively defined and beyond the scope of this document.

— Acceptance testing: the process of determining whether an implementation satisfies acceptance criteria and enables the user to determine whether or not to accept the implementation. This includes the planning and execution of several kinds of tests (e.g. functionality, quality, and speed performance testing) that demonstrate that the implementation satisfies the user requirements.

— Performance testing: measures the performance characteristics of an implementation under test (IUT) such as its throughput, responsiveness, etc. under various conditions.

— Robustness testing: the process of determining how well an implementation is able to conceal problems from attempting to reconstruct an image from an ill-formed codestream.

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/IEC 21122-1, *Information technology — JPEG XS low-latency lightweight image coding system — Part 1: Core coding system*

ISO/IEC 21122-2, *Information technology — JPEG XS low-latency lightweight image coding system — Part 2: Profiles and buffer models*

ISO/IEC 21122-3, *Information technology — JPEG XS low-latency lightweight image coding system — Part 3: Transport and container formats*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 21122-1, ISO/IEC 21122-2, ISO/IEC 21122-3 and the following 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
executable test suite
set of executable test cases in the form of codestreams that are input into an implementation under test

3.2
JPEG XS file
sequence of bytes encoding an image in the JXS file format

Note 1 to entry: This format is specified in ISO/IEC 21122-3.

3.3
procedure
set of steps which accomplishes one of the tasks which comprise an encoding or decoding process

3.4
test codestream set
(sub-)set of codestreams that are input to an implementation under test for a particular test purpose

4 Abbreviated terms

ASCII	American Standard Code for Information Interchange
ETS	executable test suite

© ISO/IEC 2022 – All rights reserved

IUT	implementation under test
PSNR	peak signal to noise ratio
TCS	test codestream set

5 Overview

The conformance files including codestreams, reference decoded images, descriptive files and auxiliary software to facilitate testing are provided at https://standards.iso.org/iso-iec/21122/-4/ed-2/en, in compressed form. File locations given in this document are expressed relative to the top level directory tree within this compressed file. A Unix style file structure and delimiters are assumed.

This document contains instructions for the use of these files.

6 General description

6.1 Overview

The ISO/IEC 21122 series, also known as JPEG XS, consists of multiple parts of which for ISO/IEC 21122-1 describes the core coding system and the syntax of a codestream, for ISO/IEC 21122-2 describes profiles and buffer models and for ISO/IEC 21122-3 describes transport and container formats. This document defines test suites for decoder conformance tests for ISO/IEC 21122-1, ISO/IEC 21122-2 and ISO/IEC 21122-3.

6.2 Codestream syntax testing

The procedures of Annex A shall be used for testing codestreams for syntactical correctness. They depend on a codestream syntax parsing tool whose source code is provided at https://standards.iso.org/iso-iec/21122/-4/ed-2/en.

6.3 Test procedures to test decoders for conformance to ISO/IEC 21122-1

The procedures defined in Annex B and the subset of ETS defined in Annex C, subclauses C.2 to C.10 that correspond to the profiles, levels and sublevels supported by the decoder, shall be used for testing decoders for compliance to ISO/IEC 21122-1. These procedures and ETS allow an IUT to evaluate conformance to ISO/IEC 21122-1 only. Annex B defines two conformance points, a strict conformance point that requires decoders to reconstruct codestreams into image data that is bitwise identical to the reference data, and a relaxed conformance point that offers additional freedom to implementations.

6.4 Test procedures to test decoders for conformance to ISO/IEC 21122-3

The procedures defined in Annex B and the subset of ETS defined in Annex C, subclause C.11 that correspond to the profiles, levels and sublevels supported by the decoder shall be used for testing decoders for compliance to ISO/IEC 21122-3. These procedures and ETS allow an IUT to evaluate conformance to ISO/IEC 21122-1 and ISO/IEC 21122-3.

6.5 File format syntax testing

The procedures that shall be used for testing JPEG XS files for compliance to the file format specified in ISO/IEC 21122-3 are defined in Annex D. They depend on a codestream syntax parsing tool whose source code is provided at https://standards.iso.org/iso-iec/21122/-4/ed-2/en.

6.6 Profile, level and sublevel conformance testing

Annex A also specifies a test procedure that shall be used for testing whether a codestream is conforming to a particular profile, level and sublevel, and whether, in particular, its coding parameters are within the constraints of the profile, level and sublevel indicated in the codestream. The test in Annex A depends on a program that is given at https://standards.iso.org/iso-iec/21122/-4/ed-2/en.

6.7 Buffer model conformance testing

Annex E specifies test procedures that shall be used for testing whether a codestream is conforming to a buffer model implied by the profile, level and sublevel indicated in the codestream. The tests in Annex E depend on a program that is given at https://standards.iso.org/iso-iec/21122/-4/ed-2/en.

6.8 Electronic attachments

Annex F lists the electronic attachments to this document and describes how to compile and use them.

7 Conformance files availability and updates

The conformance test images, streams and conformance test software released with this document are the latest tested versions available at the date at which the text was released.

© ISO/IEC 2022 – All rights reserved

Annex A
(normative)

Codestream syntax testing procedures

A.1 General

This annex defines a procedure that shall be followed for determining whether a codestream is syntactically well-formed and follows the syntactical requirements of ISO/IEC 21122-1. The test procedure also checks whether the coding parameters of a given codestream are consistent with the profile, level and sublevel indicators that are part of the picture header (see ISO/IEC 21122-1). A typical use case for this test is to check whether a given encoder generates profile, level and sublevel information correctly. To this end, a Python1) test script `jxscodestream.py` that performs a syntax analysis of a given codestream is provided at https://standards.iso.org/iso-iec/21122/-4/ed-2/en.

A.2 Installation

The test tool requires installation of a Python 2.7 interpreter on the computer system to be used for performing a test. Python is available for multiple operating systems at https://docs.python.org.

A.3 Usage of the syntax test tool

For testing a particular codestream for syntactical correctness and correct indication of profile, level and sublevel, the syntax analyser `jxscodestream.py` tool shall be run on a command line as follows:

```
jxscodestream.py codestream.jxs
```

where `codestream.jxs` is the codestream to be tested. A codestream is not conforming in case the above tool reports any error.

The lack of detection of any conformance violation by the syntax test tool should not be considered as a definite proof that the codestream under testing conforms to all constraints required for conformance to ISO/IEC 21122-1 and ISO/IEC 21122-2. ISO/IEC 21122-5 provides additional means for testing a codestream for conformance to the ISO/IEC 21122 series by feeding it into the reference software implementation.

1) Python is a trademark of the Python Software Foundation. This information is given for the convenience of users of this document and does not constitute an endorsement by ISO/IEC of the product named. Equivalent products may be used if they can be shown to lead to the same results.

© ISO/IEC 2022 – All rights reserved

Annex B
(normative)

Decoder testing procedures

B.1 General

This annex defines procedures that shall be followed for determining whether a JPEG XS decoder implementation is conforming to a particular profile, level and sublevel. It defines two conformance points; strict conformance and relaxed conformance.

B.2 Decoder test procedure

The following steps shall be performed for the purpose of testing a decoder implementation:

- Select to test for ISO/IEC 21122-1 only or ISO/IEC 21122-1 and ISO/IEC 21122-3.
- Select a profile, level and sublevel to test against.
- The profile, level and sublevel define a TCS (test codestream set), which consists of codestreams or files within the test suite that conform to
  - the codestream syntax specified in ISO/IEC 21122-1 or the file format specified in ISO/IEC 21122-3,
  - the selected profile,
  - a level whose sample count and sample rate is smaller than or equal to the sample count and sample rate of the selected level, and
  - to a sublevel whose bitrate is smaller or equal than the bitrate of the selected sublevel.
- Decode each codestream or file using the implementation under test.
- The decoded outputs are format converted if necessary.
- The difference between the decoded outputs and reference outputs is measured.
- If the decoded and reference are identical, the test proceeds to the next codestream in the TCS.
- The decoded image shall be compared to the reference image. Depending on the outcome of this test, the decoder is conforming to the strict conformance point, or only to the relaxed conformance point, or is non-conforming as follows:
  - Strict Conformance Point: If the decoded images are identical (i.e. the PSNR difference is infinity) to the reference images for all codestreams in the TCS, the decoder conforms to the strict conformance point in the given profile, level and sublevel.
  - Relaxed Conformance Point: If the PSNR difference between all decoded image and the reference image is above or equal to the PSNR bounds listed in Annex C for all codestreams in the TCS, the decoder conforms to the relaxed conformance point in the given profile, level and sublevel.
  - If for at least one codestream in the TCS the PSNR difference between the decoded image and the reference image is below the PSNR bounds listed in Annex C for this codestream, the decoder is non-conforming to the given profile, level and sublevel.

© ISO/IEC 2022 – All rights reserved

The entire set of test codestreams along with PSNR bounds is defined in more detail in Annex C. Instructions how to measure PSNR are specified in subclause B.8.

B.3 Files for testing

A particular ETS defines the input codestreams and PSNR bounds. These are specified in Annex C for all profiles.

B.4 Decoder settings

Decoders may have mechanisms for supporting various decompression settings. These may be set in the most advantageous way to achieve strict or relaxed conformance. For example, a decoder with a "fast mode" and an "accurate mode" may be set to the "accurate mode" to determine the level of conformance. These settings should be noted in any statement of conformance. Settings that allow the output resolution or spatial region of the reference decoded images to be matched may be changed for each decoded image. The same user-controlled settings for accuracy shall be used for all test codestreams of a TCS.

B.5 Output file format conversion

The reference decoded images are provided in a specific file format defined in subclause B.7. In order to compare decoded images from the decoder under test with these images, several conversions may be necessary. These conversions may be done as post-processing steps outside of the decoder solely for determining conformance. There is no requirement for a conforming decoder to perform these processes as part of its normal operation. These conversions shall not introduce a quality change (either loss or gain) except as required by the specific conversions described in subclauses B.6 and B.7.

B.6 Sample format conversion

Image sample values are always unsigned integers. For the purpose of testing, it is of advantage to represent the integer output of the IUT in the pgx format defined in subclause B.7 as ISO/IEC provides tools to measure on such files directly. Any up-sampling or colour space format conversions shall be disabled for the purpose of testing should a decoder offer these as optional features.

There is no requirement that a conforming decoder has to generate output to the specified format, and the representation of the output of the IUT in this format only facilitates the testing process. That is, it is acceptable to include additional lossless format conversions in the testing procedure provided the comparison between reconstructed and reference images is performed in the normative way.

Conversion from YC_bC_r to RGB, colour space conversions, up-sampling or interpolation of components to populate the entire sample grid may be provided as optional features of a decoder implementation and need to be disabled for testing purposes. This conversion between colour spaces is not related to the RCT or Star-Tetrix transformations that are normatively defined in ISO/IEC 21122-1, both of which shall be executed for the purpose of reference testing if enabled in the codestream. Conformance testing applies to the reconstructed sample values only, bare any interpretation relative to a colour space, and no colour space conversions are made for the purpose of testing. In particular, colour filter array images are understood as four-component images consisting of one red, two green and one blue component, consistent with their interpretation in ISO/IEC 21122-1. The width of their sampling grid is half the width of the sensor array, and the height of their sensor grid is half the height of the sensor array used to create these images.

B.7 Reference components file format

B.7.1 General

This subclause specifies the file format, called pgx, of the reference images used for comparison with the output of the decoder under test. The decoder under test is not required to produce this particular file format, though it is advantageous to perform a conversion to this file format for testing purposes as ISO/IEC provides test tools that are able to decode this format. Any necessary lossless conversion to this format, as specified by subclause B.5, may be applied as part of the test procedure.

The format consists of a directory file, one header file per component, and one raw data file per component.

NOTE The pgx directory file does not separate lines by CR/LF pairs, i.e. the ISO/IEC 646 code sequence 13, 10 / hex 0x0d 0x0a.

B.7.2 Directory file format

The directory file is the file that is provided as input file to the comparison tool. It consists, for each component, of the file name of the raw data file encoded in ISO/IEC 646 (ASCII), relative to the path where the directory file is located. Each raw data file name is terminated by a single line feed character (ISO/IEC 646 code 10, hex 0x0a).

NOTE The pgx directory file does not separate lines by CR/LF pairs, i.e. the ISO/IEC 646 code sequence 13, 10 / hex 0x0d 0x0a.

B.7.3 Header file format

The header file is derived from the file name of the raw data file listed in the directory file by removing the postfix `.raw` and replacing it by the postfix `.h`. It describes the format in which the raw format is encoded. There is one separate header file per component.

Each header file consists of a single line, terminated by a single ISO/IEC 646 line feed character (code 10, hex 0x0a) describing the format. It consists of the following fields, where angle brackets < > indicate parameters described below and `SPC` indicates an ISO/IEC 646 blank space (code 32, hex 0x20):

```
P<data format>SPC<endianness>SPC<signedness><precision>SPC<width>SPC<height>
```

where

`P`	identifies the header file and shall be present. It has no particular meaning beyond format identification.
`<data format>`	identifies the sample format. ISO/IEC 21122 stores integer samples only, which are indicated by the single character `G` as `data format`.
`<endianness>`	identifies the endianness of the encoded data. The character sequence `ML` indicates big endian encoding, i.e. most significant byte first, the character sequence `LM` little endian encoding, i.e. least significant byte first.
`<signedness>`	indicates whether sample values are signed or unsigned. ISO/IEC 21122 covers only unsigned samples, indicated by the character `+` in this field.
`<precision>`	indicates the bit-precision of the sample values in the component described by this header file. The bit-precision is represented as ISO/IEC 646 encoded decimal number.
`<width>`	is the number of samples per line for this component, represented as ISO/IEC 646 encoded decimal number.

© ISO/IEC 2022 – All rights reserved

`<height>`	is the number of lines of the image, represented as ISO/IEC 646 encoded decimal number.

B.7.4 Data file format

Each data file contains the sample values themselves in raster scan order, left to right, top to bottom. If the precision of the component is 8 or below, each sample is represented in 8 bits, right aligned to the entire byte, i.e. unused bits remain 0 and make up the most-significant bits of the byte. If the precision is larger than 8, each sample is represented by two bytes, encoded in the order indicated by the header, i.e. either most significant byte first if the endianness field is `ML`, or with the least significant byte first if the endianness field is `LM`. The data bits are right-aligned into the two bytes, most significant bits remain 0 if necessary.

B.8 Comparison of decoded and formatted components with reference components

The image reconstructed from a TCS and the reference test image for the TCS need to be identical sample by sample in order to match the strict conformance point. To test for the relaxed conformance, the PSNR (Peak Signal to Noise Ratio) between the reconstructed image and reference image shall be computed as follows:

$$\mathrm{PSNR} = -10\log_{10}\left(\frac{1}{N_\mathrm{c}}\sum_{c=0}^{N_\mathrm{c}-1}\frac{\sum_{x=0}^{W_\mathrm{f}-1}\sum_{y=0}^{H_\mathrm{f}-1}\chi_{(x,y,c)}\left(D_{(x,y,c)}-R_{(x,y,c)}\right)^2}{\left(2^{B[c]}-1\right)^2\sum_{x=0}^{W_\mathrm{f}-1}\sum_{y=0}^{H_\mathrm{f}-1}\chi_{(x,y,c)}}\right)$$

where N_c is the number of components in the image, W_f the width of the sensor grid, H_f the height of the sensor grid, $\chi_{(\mathrm{x,y,c})}$ the indicator function that is 1 if the position (x,y) of the sampling grid is populated by component c, $D(x,y,c)$ is the sample value of component c of the decoded image output by the IUT at sample grid position (x,y), $R(x,y,c)$ the sample value of component c of the reference image at sample grid position (x,y) and $B[c]$ the bit-precision of component c as found in the pgx header file.

NOTE 1 By this formula, the PSNR is computed from the mean square error over all populated sample grid positions.

NOTE 2 The `difftest_ng` test tool available as an electronic attachment to this document can be used to implement such a test. `difftest_ng` reports a peak signal to noise ratio (PSNR) of INF in the case that decoded and reference image are identical. Otherwise, it reports the PSNR value as computed by the above formula.

NOTE 3 The reference image is the expected output of a strictly conforming decoder implementation; thus, the PSNR bound is a bound of the distortion between the output of a strictly conforming decoder, and a decoder conforming to the relaxed conformance point. In particular, the PSNR bound should **not** be confused with the PSNR between an original and a reconstructed image.

Annex C
(normative)

Decoder conformance tests

C.1 General

This annex specifies the abstract test suites and executable test suites that shall be used in the conformance test procedures from Annex B. References reconstructed images are given by substituting the suffix `.jxs` by `.pgx`. If the column indicating the PSNR bound indicates a dash ("-"), the corresponding codestream shall not be included for testing an IUT for the relaxed conformance point. If the PSNR bound indicates "INF", a decoder is required to match the reference output bitwise-exact, regardless the conformance point being tested.

C.2 Reference codestreams for the light subline 422.10 profile

Table C.1 lists the codestreams for testing implementations against the light subline 422.10 profile.

Table C.1 — Light subline 422.10 profile test streams

TCS	Level	Sublevel	PSNR bound
`2.jxs`	4k-2	3bpp	66
`48.jxs`	4k-2	3bpp	67
`20.jxs`	unrestricted	9bpp	67
`28.jxs`	4k-1	unrestricted	60
`50.jxs`	4k-1	unrestricted	60
`36.jxs`	2k-1	unrestricted	93

C.3 Reference codestreams for the light 422.10 profile

Table C.2 lists the codestreams for testing implementations against the light 422.10 profile.

Table C.2 — Light 422.10 profile test streams

TCS	Level	Sublevel	PSNR bound
`3.jxs`	4k-2	6bpp	60
`49.jxs`	4k-2	3bpp	60
`12.jxs`	4k-2	3bpp	60
`51.jxs`	4k-2	6bpp	60
`21.jxs`	unrestricted	12bpp	67
`29.jxs`	4k-1	12bpp	60
`37.jxs`	2k-1	9bpp	84

C.4 Reference codestreams for the light 444.12 profile

Table C.3 lists the codestreams for testing implementations against the light 444.12 profile.

© ISO/IEC 2022 – All rights reserved

Table C.3 — Light 444.12 profile test streams

TCS	Level	Sublevel	PSNR bound
`4.jxs`	4k-2	3bpp	72
`52.jxs`	4k-2	6bpp	80
`13.jxs`	4k-1	9bpp	72
`22.jxs`	unrestricted	12bpp	69
`53.jxs`	unrestricted	12bpp	-
`30.jxs`	4k-1	12bpp	72
`38.jxs`	2k-1	unrestricted	73

C.5 Reference codestreams for the main 422.10 profile

Table C.4 lists the codestreams for testing implementations against the main 422.10 profile.

Table C.4 — Main 422.10 profile test streams

TCS	Level	Sublevel	PSNR bound
`5.jxs`	4k-2	3bpp	65
`54.jxs`	4k-2	6bpp	64
`14.jxs`	4k-2	9bpp	75
`23.jxs`	unrestricted	12bpp	-
`31.jxs`	4k-1	12bpp	60
`39.jxs`	2k-1	9bpp	74
`66.jxs`	2k-1	12bpp	66

C.6 Reference codestreams for the main 444.12 profile

Table C.5 lists the codestreams for testing implementations against the main 444.12 profile.

Table C.5 — Main 444.12 profile test streams

TCS	Level	Sublevel	PSNR bound
`6.jxs`	4k-2	3bpp	67
`55.jxs`	4k-2	6bpp	64
`11.jxs`	4k-2	6bpp	80
`15.jxs`	4k-1	9bpp	78
`24.jxs`	unrestricted	12bpp	68
`56.jxs`	unrestricted	12bpp	-
`32.jxs`	4k-1	12bpp	72
`40.jxs`	2k-1	unrestricted	72

C.7 Reference codestreams for the main 4444.12 profile

Table C.6 lists the codestreams for testing implementations against the main 4444.12 profile.

Table C.6 — Main 4444.12 profile test streams

TCS	Level	Sublevel	PSNR bound
`8.jxs`	4k-2	3bpp	65
`57.jxs`	4k-2	6bpp	64
`17.jxs`	4k-1	9bpp	78
`26.jxs`	unrestricted	12bpp	-
`58.jxs`	unrestricted	12bpp	67
`34.jxs`	4k-1	12bpp	72
`42.jxs`	2k-1	9bpp	63
`44.jxs`	8k-1	9bpp	-
`59.jxs`	4k-1	12bpp	66
`46.jxs`	4k-1	12bpp	-

C.8 Reference codestreams for the high 444.12 profile

Table C.7 lists the codestreams for testing implementations against the high 444.12 profile.

Table C.7 — High 444.12 profile test streams

TCS	Level	Sublevel	PSNR bound
`7.jxs`	4k-2	3bpp	62
`60.jxs`	4k-2	6bpp	62
`16.jxs`	4k-1	9bpp	78
`25.jxs`	unrestricted	12bpp	66
`61.jxs`	unrestricted	12bpp	-
`33.jxs`	4k-1	12bpp	-
`41.jxs`	2k-1	unrestricted	72

C.9 Reference codestreams for the high 4444.12 profile

Table C.8 lists the codestreams for testing implementations against the high 4444.12 profile.

Table C.8 — High 4444.12 profile test streams

TCS	Level	Sublevel	PSNR bound
`9.jxs`	4k-2	3bpp	62
`62.jxs`	4k-2	6bpp	62
`18.jxs`	4k-1	9bpp	78
`27.jxs`	unrestricted	12bpp	66
`63.jxs`	unrestricted	12bpp	-
`35.jxs`	4k-1	12bpp	72
`43.jxs`	2k-1	9bpp	62
`45.jxs`	8k-1	6bpp	-
`47.jxs`	4k-1	12bpp	66
`64.jxs`	4k-1	12bpp	-

© ISO/IEC 2022 – All rights reserved

C.10 Reference codestreams for the unrestricted profile

Table C.9 lists the codestreams for testing implementations against the unrestricted profile.

Table C.9 — Unrestricted profile test streams

TCS	Level	Sublevel	PSNR bound
`10.jxs`	unrestricted	3bpp	70
`65.jxs`	unrestricted	6bpp	69
`19.jxs`	4K-1	9bpp	78
`201.jxs`	2k-1	3bpp	59
`203.jxs`	1k-1	3bpp	67

C.11 Reference files for file format conformance testing

Table C.10 lists the files for testing implementations against the file format specified in ISO/IEC 21122-3 and the main 444.12 profile.

Table C.10 — File format profile test streams

TCS	Level	Sublevel	PSNR bound
`1.jxs`	4k-2	3bpp	60

C.12 Reference files for the main 420.12 profile

Table C.11 lists the files for testing implementations against the main 420.12 profile.

Table C.11 — main 420.12 profile test streams

TCS	Level	Sublevel	PSNR bound
`200.jxs`	2k-1	3bpp	60
`202.jxs`	2k-1	3bpp	68

C.13 Reference files for the MLS.12 profile

Table C.12 lists the files for testing implementations against the MLS.12 profile.

Table C.12 — MLS.12 profile test streams

TCS	Level	Sublevel	PSNR bound
`204.jxs`	1k-1	full	INF
`205.jxs`	1k-1	full	INF
`209.jxs`	1k-1	full	INF
`217.jxs`	1k-1	full	INF
`218.jxs`	1k-1	full	INF

C.14 Reference files for the LightBayer profile

Table C.13 lists the files for testing implementations against the LightBayer profile.

Table C.13 — LightBayer profile test streams

TCS	Level	Sublevel	PSNR bound
`210.jxs`	1k-1	3bpp	73
`211.jxs`	1k-1	3bpp	70
`212.jxs`	1k-1	3bpp	67

C.15 Reference files for the MainBayer profile

Table C.14 lists the files for testing implementations against the MainBayer profile.

Table C.14 — MainBayer profile test streams

TCS	Level	Sublevel	PSNR bound
`213.jxs`	1k-1	3bpp	66
`214.jxs`	1k-1	3bpp	70
`216.jxs`	1k-1	3bpp	-

C.16 Reference files for the HighBayer profile

Table C.15 lists the files for testing implementations against the HighBayer profile.

Table C.15 — HighBayer profile test streams

TCS	Level	Sublevel	PSNR bound
`215.jxs`	1k-1	3bpp	65

© ISO/IEC 2022 – All rights reserved

Annex D
(normative)

File format syntax testing procedures

D.1 General

This annex defines procedures that shall be followed to determine whether a JXS file is syntactically well-formed and follows the syntactical requirements of ISO/IEC 21122-3. To this end, a Python test script `jp2file.py` has been attached to this document that performs a syntax analysis of a given codestream.

D.2 Installation

The test tool is designed for a Python 2.7 interpreter . Python is available for multiple operating systems.

D.3 Usage of the syntax test tool

To test a particular JXS file for syntactical correctness, the syntax analyser shall be run from a command line as follows:

```
jp2file.py jxsfile.jxs
```

where `jxsfile.jxs` is the file to be tested. A file fails the conformance test if the above syntax analysis tool reports any error.

The lack of detection of any conformance violation by the syntax test tool should not be considered as a definite proof that the file under test conforms to all constraints required for conformance to ISO/IEC 21122-3.

Annex E
(normative)

Buffer model conformance testing

E.1 General

This annex defines a method that shall be used to test whether a particular codestream conforms to the JPEG XS buffer model specified in ISO/IEC 21122-2, and hence if a JPEG XS decoder with a smoothing buffer sized according to ISO/IEC 21122-2 is able to receive and decode the codestream under test without running into a buffer underflow or overflow. This test is also useful to test encoder implementations, as to ensure whether the generated codestreams are satisfying the buffer model requirements of ISO/IEC 21122-2. The test itself is implemented by a Python program `bufferModelChecker.py` that is available at https://standards.iso.org/iso-iec/21122/-4/ed-2/en.

E.2 Installation

The test depends on the availability of a Python 3 interpreter on the computer. In addition, the test depends on a suitable decoder implementation that extracts buffer information from the codestream. Python is available for multiple operating systems.

NOTE The JPEG XS reference software contained in ISO/IEC 21122-5 implements the required feature extraction and can therefore be used to obtain this information.

E.3 Using the buffer model test tool

E.3.1 General

Testing a codestream for conformance to the JPEG XS buffer model involves two steps: first, the extraction of codestream fragment sizes and number of coding groups from the codestream under test, and second, the analysis of the data generated by the first step with the buffer model checker.

E.3.2 Extraction of codestream fragment sizes and code group counts

The buffer model test tool requires as input information on the codestream fragments represented by the codestream under test. For the purpose of buffer model testing, the following data shall be extracted from the codestream:

— a contiguous counter, starting from 0, that enumerates all codestream fragments;

— the size of the codestream fragment in bits;

— the number of code groups encoded by the packet that is included in the codestream fragment.

For the definition of packets and code groups, see ISO/IEC 21122-1. For the definition of codestream fragments, see ISO/IEC 21122-2.

The data on codestream fragments and code group counts shall then be formatted in a file that contains for each codestream fragment a line as follows:

```
<counter>;<size>;<code groups>
```

© ISO/IEC 2022 – All rights reserved

where `<counter>` enumerates the codestream fragments in increasing order, starting from 0, `<size>` is the size of the codestream fragment in bits, and `<code groups>` is the number of code groups in the packet. This file is input to the buffer model checker described in the following subclause.

NOTE The JPEG XS Reference software contained in ISO/IEC 21122-5 can be used to extract this information from a given codestream Codestream fragment information is generated by the reference software by running it as follows:

```
jxs_decoder –F fragfile.csv input.jxs output.pgx
```

where `fragfile.csv` will contain the fragment information as required, `input.jxs` is the JPEG XS codestream to analyse and `output.pgx` will contain the reconstructed image.

E.3.3 Testing codestream fragment information

In the second step, the output of the codestream analysis shall be fed into the buffer model checker which will perform the conformance check. For this, `bufferModelChecker.py` shall be run on the command line as follows:

```
bufferModelChecker.py –-profile <profile> --mainlevel <level> --sublevel <sublevel>
--colorFormat <format> --type <buffermodeltype> --imageWidth <width>  [--jxs <codestream>]
<fragmentinfo>
```

where the parameters are defined as follows:

`<profile>`	is the profile the codestream claims to be conforming to. Possible values are: `Main444.10, Main444.12, High444.12, Main4444.12, Light444.12, LightSubline422.10, High4444.12, Light422.10, Main420.12, LightBayer, MainBayer, HighBayer` NOTE: The buffer model checker does not accept `MLS.12` as profile because this profile does not imply a buffer mode.
`<level>`	is the level the codestream claims to be conforming to. Possible values are: `1k-1, 2k-1, 4k-1, 4k-2, 4k-3, 8k-1, 8k-2, 8k-3, 10k-1`
`<sublevel>`	is the sublevel the codestream claims to be conforming to. Possible values are: `Full, Sublev12bpp, Sublev9bpp, Sublev6bpp, Sublev3bpp, Sublev2bpp`
`<format>`	is the sampling format of the input image. Possible values are: `400, 420, 422, 444, 4224, 4444, CFA`
`<buffermodeltype>`	is the type of the buffer model to be assumed. The following values are accepted: 1: a constant bitrate smoothing buffer with limited transmission latency is assumed; 2: a constant bitrate buffer with variable transmission latency is assumed. NOTE: The buffer model type 0 does not imply a buffer, and thus does not require checking by this tool.
`<jxsfile>`	is, optionally, the input codestream itself. If given, a file size check against sublevel constraints is performed.
`<fragmentinfo>`	is the file generated in the first step containing fragment index, fragment bit size and number of included code groups, separated by semicolons.
`<width>`	is the width of the image in sample grid points.

The output of the buffer model checker is a JSON-encoded line as follows:

```
{'BM_Result'      : <ok>}
```

This output encodes whether the codestream is acceptable for the selected buffer model. The parameter `<ok>` is `True` in case it is, and `False` in case the conformance test fails.

© ISO/IEC 2022 – All rights reserved

Annex F
(informative)

Attached software

F.1 General

This annex lists the software available from https://standards.iso.org/iso-iec/21122/-4/ed-2/en and lists, if necessary, how to compile and install them.

F.2 List of attachments

Table F.1 lists attached software, their function, and where compilation instructions are found.

Table F.1 — List of attached software

Name	Function	Compilation/installation instructions
`difftest_ng`	Error measurement and format conversion	Subclause F.4
`jxscodestream.py`	Tests an ISO/IEC 21122-1 code-stream for syntactical correctness and for the correctness of the profile, level and sublevel indicator.	Subclause F.3
`jp2file.py`	Tests an ISO/IEC 21122-3 file for syntactical correctness.	Subclause F.3
`bufferModelChecker.py`	Tests whether a codestream is decodable by a decoder implementing a smoothing buffer operating and sized as specified in ISO/IEC 21122-2.	Subclause F.3

F.3 Installation of Python scripts

The Python programming language is an interpreted language and programs implemented in Python do not require any compilation. Once successfully installed, Python programs can be run immediately from the command line of the operating system. Care should be taken on the particular version of the Python interpreter as older versions such as Python 2.7 are not compatible to newer versions such as Python 3.

NOTE Testing as described in Annex D requires a different version of the Python language than the testing protocol described in Annex E.

F.4 Compilation of the difftest tool

Compilation of `difftest_ng` can be performed in the presence of the `gcc/g++` compiler suite, the GNU `make` utility and a `bash`-compatible command line. In the presence of these utilities, compilation of `difftest_ng` can be performed by the following steps: First configure the compilation environment by entering

```
./configure
```

in the `difftest_ng` directory. This will detect compiler specifics and the available software packages on the target infrastructure. Afterwards, `difftest_ng` is compiled by entering

```
make
```

on the same directory.

For Windows^{TM}[2)] operating systems, a Visual Studio^{TM} configuration file is included as well. This configuration file needs to be loaded, and compilation is immediate through the "Build" menu of this compiler suite.

2) Windows and Visual Studio are trademarks of Microsoft®. This information is given for the convenience of users of this document and does not constitute an endorsement by ISO/IEC of the products named. Equivalent products may be used if they can be shown to lead to the same results.

© ISO/IEC 2022 – All rights reserved

Bibliography

[1] ISO/IEC 646, *Information technology — ISO 7-bit coded character set for information interchange*

[2] ISO/IEC 21122-5, *Information technology — JPEG XS Low-latency lightweight image coding system — Part 5: Reference software*

ICS 35.040.30

Price based on 21 pages

© ISO/IEC 2022 – All rights reserved