INTERNATIONAL STANDARD

ISO/IEC 15444-15

First edition
2019-10

Information technology — JPEG 2000 image coding system —

Part 15: High-Throughput JPEG 2000

Technologies de l'information — Système de codage d'images JPEG 2000 —

Partie 15: JPEG 2000 à haut débit

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

Reference number
ISO/IEC 15444-15:2019(E)

© ISO/IEC 2019

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2019

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
Fax: +41 22 749 09 47
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

© ISO/IEC 2019 – All rights reserved

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).

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 http://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.

This document was prepared by ITU-T as ITU-T T.814 (06/2019) and drafted in accordance with its editorial rules. It was assigned to 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 15444 series can be found on the ISO website.

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.

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

INTERNATIONAL STANDARD ISO/IEC 15444-15
RECOMMENDATION ITU-T T.814

Information technology – JPEG 2000 image coding system: High-throughput JPEG 2000

Summary

The computational complexity of the block-coding algorithm of Rec. ITU-T T.800 | ISO/IEC 15444-1 can be a challenge in some applications.

Rec. ITU-T T.814 | ISO/IEC 15444-15 specifies a high-throughput (HT) block-coding algorithm that can be used in place of the block-coding algorithm specified in Rec. ITU-T T.800 | ISO/IEC 15444-1.

The HT block-coding algorithm increases decoding and encoding throughput and allows mathematically lossless transcoding to and from the block-coding algorithm specified in Rec. ITU-T T.800 | ISO/IEC 15444-1. This is achieved at the expense of some loss in coding efficiency and substantial elimination of quality scalability.

The HT block-coding algorithm adopts a coding pass structure like that of the block-coding algorithm of Rec. ITU-T T.800 | ISO/IEC 15444-1. No more than three coding passes are required for any given code-block in the final codestream, and arithmetic coding is replaced with a combination of variable length coding tools, adaptive run-length coding and simple bit-packing. The algorithm involves three passes: a significance propagation pass (HT SigProp coding pass), a magnitude refinement pass (HT MagRef coding pass) and a cleanup pass (HT cleanup coding pass).

The HT MagRef coding pass is identical to that of the block-coding algorithm of Rec. ITU-T T.800 | ISO/IEC 15444-1, operating in the bypass mode, except that code bits are packed into bytes with a little-endian bit order. That is, the first code bit in a byte appears in its LSB, as opposed to its MSB.

The HT SigProp coding pass is also very similar to that of the block-coding algorithm of Rec. ITU-T T.800 | ISO/IEC 15444-1, operating in the BYPASS mode, with the following two differences:

- code bits are again packed into bytes of the raw bit-stream with a little-endian bit order, instead of big-endian bit packing order; and
- the significance bits associated with a set of four stripe columns are emitted first, followed by the associated sign bits, before advancing to the next set of stripe columns, instead of inserting any required sign bit immediately after the same sample's magnitude bit.

The HT cleanup coding pass is, however, significantly different from that of the block-coding algorithm of Rec. ITU-T T.800 | ISO/IEC 15444-1, and most of ITU-T T.814 | ISO/IEC 15444-15 is devoted to its description.

Aside from the block-coding algorithm itself and the parsing of packet headers, the HT block-coding algorithm preserves the syntax and semantics of other parts of the codestream specified in Rec. ITU-T T.800 | ISO/IEC 15444-1.

Recommendation ITU-T T.814 (2019) is a common text with ISO/IEC 15444-15:2019, both in their first edition.

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

History

Edition	Recommendation	Approval	Study Group	Unique ID*
1.0	ITU-T T.814	2019-06-13	16	11.1002/1000/13912

* To access the Recommendation, type the URL http://handle.itu.int/ in the address field of your web browser, followed by the Recommendation's unique ID. For example, http://handle.itu.int/11.1002/1000/11830-en.

© ISO/IEC 2019 – All rights reserved

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications, information and communication technologies (ICTs). The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

Compliance with this Recommendation is voluntary. However, the Recommendation may contain certain mandatory provisions (to ensure, e.g., interoperability or applicability) and compliance with the Recommendation is achieved when all of these mandatory provisions are met. The words "shall" or some other obligatory language such as "must" and the negative equivalents are used to express requirements. The use of such words does not suggest that compliance with the Recommendation is required of any party.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU had received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementers are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database at http://www.itu.int/ITU-T/ipr/.

© ITU 2019

All rights reserved. No part of this publication may be reproduced, by any means whatsoever, without the prior written permission of ITU.

© ISO/IEC 2019 – All rights reserved

CONTENTS

Page

1 Scope 1

2 Normative references 1

3 Terms and definitions 1

4 Abbreviations and symbols 1

5 Convention 2

6 Conformance 2
- 6.1 HTJ2K codestream 2
- 6.2 HTJ2K decoding algorithm 2
- 6.3 JPH file 3

7 HT block decoding algorithm 3
- 7.1 Retrieving bit-streams from HT segments 3
- 7.2 Quad-based scanning pattern 9
- 7.3 HT Cleanup decoding algorithm 10
- 7.4 HT SigProp decoding procedure 20
- 7.5 HT MagRef decoding procedure 22
- 7.6 Sample output values 22

8 Constrained codestream sets 23
- 8.1 Overview 23
- 8.2 HTONLY, HTDECLARED and MIXED sets 23
- 8.3 SINGLEHT and MULTIHT sets 23
- 8.4 RGN and RGNFREE sets 23
- 8.5 HOMOGENEOUS and HETEROGENEOUS sets 23
- 8.6 LOCAL and FRAG sets 24
- 8.7 Bounded magnitude sets 24
- 8.8 CPF_N sets 25

9 Media types 26

Annex A (normative) – HTJ2K codestream syntax 27
- A.1 General 27
- A.2 SIZ marker segment 27
- A.3 CAP marker segment 27
- A.4 COD and COC marker segments 29
- A.5 RGN marker segment 30
- A.6 Corresponding Profile (CPF) marker segment 30

Annex B (normative) – HT data organization 32
- B.1 HT Sets 32
- B.2 HT segments 32
- B.3 Packets, Z_blk and S_blk 32

Annex C (normative) – CxtVLC tables 34

Annex D (normative) – JPH file format 54
- D.1 General 54
- D.2 JP2 Header box 54
- D.3 File Type box 54
- D.4 Colour Specification box 54
- D.5 Contiguous Codestream box 55
- D.6 Channel Definition box 55

Annex E (normative) – Media type specifications and registrations 57
- E.1 General 57
- E.2 JPH file 57
- E.3 Single HTJ2K codestream 57

Annex F (informative) – HT block encoding procedures 59
- F.1 Overview 59

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

F.2 Bit-planes, exponents, MagSgn values and EMB patterns 61
F.3 Cleanup pass encoding steps 62
F.4 Bit-stuffing and byte-stream termination procedures 65

Bibliography 72

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

INTERNATIONAL STANDARD
ITU-T RECOMMENDATION

Information technology – JPEG 2000 image coding system: High-throughput JPEG 2000

1 Scope

This Recommendation | International Standard specifies an alternate block-coding algorithm that can be used in place of the block-coding algorithm specified in Rec. ITU-T T.800 | ISO/IEC 15444-1. This alternate block-coding algorithm offers a significant increase in throughput at the expense of slightly reduced coding efficiency, while a) allowing mathematically lossless transcoding to and from codestreams that use the block-coding algorithm specified in Rec. ITU-T T.800 | ISO/IEC 15444-1, and b) preserving codestream syntax and features specified in Rec. ITU-T T.800 | ISO/IEC 15444-1.

Recommendation ITU-T T.814 (2019) is a common text with ISO/IEC 15444-15:2019, both in their first edition.

2 Normative references

The following Recommendations and International Standards contain provisions which, through reference in this text, constitute provisions of this Recommendation | International Standard. At the time of publication, the editions indicated were valid. All Recommendations and Standards are subject to revision, and parties to agreements based on this Recommendation | International Standard are encouraged to investigate the possibility of applying the most recent edition of the Recommendations and Standards listed below. Members of IEC and ISO maintain registers of currently valid International Standards. The Telecommunication Standardization Bureau of the ITU maintains a list of currently valid ITU-T Recommendations.

2.1 Identical Recommendations | International Standards

– Recommendation ITU-T T.800 (2019) | ISO/IEC 15444-1:2019, *Information technology – JPEG 2000 image coding system: Core coding system*.

2.2 Paired Recommendations | International Standards equivalent in technical content

– Recommendation ITU-T H.273 (2016), *Coding-independent code points for video signal type identification*.

– ISO/IEC 23001-8:2016, *Information technology – MPEG systems technologies – Part 8: Coding-independent code points*.

2.3 Additional references

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

3 Terms and definitions

For the purposes of this Recommendation | International Standard, the terms and definitions given in Rec. ITU-T T.800 | ISO/IEC 15444-1 apply.

4 Abbreviations

For the purposes of this Recommendation | International Standard, the abbreviations and symbols defined in Rec. ITU-T T.800 | ISO/IEC 15444-1 and the following apply.

AZC	All Zero Context
CUP	Cleanup coding Pass
CPF	Corresponding Profile
CxtVLC	Context adaptive Variable Length Code
EMB	Exponent Max Bound
FRAG	Fragmented
HT	High-Throughput
HTJ2K	High-Throughput JPEG 2000

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

HTIRV	High-Throughput Irreversible
HTREV	High-Throughput Reversible
LSB	Least-Significant Bit
MAGB	Magnitude Bound
MagRef	Magnitude Refinement
MagSgn	Magnitude and Sign
MRP	MagRef coding pass
MSB	Most-Significant Bit
SigProp	Significance Propagation
U-VLC	Unsigned residual VLC
VLC	Variable Length Coding

5 Conventions and symbols

For the purposes of this Recommendation | International Standard, the symbols defined in Rec. ITU-T T.800 | ISO/IEC 15444-1 and the following apply:

Dcup[n]	Byte n of an HT Cleanup segment
Dref[n]	Byte n of an HT Refinement segment
Hblk	Height of a code-block, measured in samples
Lcup	Length in bytes of HT Cleanup segment
Lref	Length in bytes of HT Refinement segment
MEL	Adaptive run-length coding algorithm
MEL_E	MEL Exponent Table
Pcup	HT Cleanup segment prefix length
QH	Height of a code-block, measured in quads
QW	Width of a code-block, measured in quads
Scup	HT Cleanup segment suffix length
SPP	HT SigProp coding Pass
u_ext	U-VLC extension component
u_pfx	U-VLC prefix component
u_sfx	U-VLC suffix component
Wblk	Width of a code-block, measured in samples
Zblk	Number of passes that can be processed within an HT Set

6 Conformance

6.1 HTJ2K codestream

A high-throughput JPEG 2000 (HTJ2K) codestream shall conform to Annex A.

6.2 HTJ2K decoding algorithm

The HTJ2K decoding algorithm processes an HTJ2K codestream as specified in Rec. ITU-T T.800 | ISO/IEC 15444-1 together with any additional signalled capability, with the exception of HT code-blocks, as defined in Annex B, in which case the following applies:

- the HT code-blocks are processed according to clause 7; and

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

- the resulting number of magnitude bits N_b, the magnitude bits $\mathrm{MSB}_i(b)$and the sign bits s_b are processed according to Rec. ITU-T T.800 | ISO/IEC 15444-1, together with any additional signalled capability.

NOTE 1 – If the two most significant bits of Ccap[15] are 0 for a codestream, all code-blocks are HT code-blocks and the decoding procedures defined in Annexes C and D of Rec. ITU-T T.800 | ISO/IEC 15444-1 are not used.

NOTE 2 – The processing of HT code-blocks specified herein is compatible with the additional capabilities specified in Rec. ITU-T T.801 | ISO/IEC 15444-2.

NOTE 3 – The symbols N_b, $MSB_i(b)$ and s_b are defined in Rec. ITU-T T.800 | ISO/IEC 15444-1.

6.3 JPH file

A JPH file shall conform to Annex D.

7 HT block-decoding algorithm

7.1 Retrieving bit-streams from HT segments

7.1.1 General

This clause specifies the process for extracting bit-streams from an HT set and its associated parameters `Z_blk` and `S_blk`, as defined in Annex B.

If `Z_blk` equals `0`, no HT segments are available for the code-block, and so all sample output values for the block shall be 0.

There are at most two HT segments available to the HT block-decoding algorithm:

- The HT cleanup segment holds the coded bytes belonging to the HT cleanup coding pass (CUP);
- The HT refinement segment holds the coded bytes belonging to the HT significance propagation (SigProp) coding pass and, optionally, an HT magnitude refinement (MagRef) coding pass. The HT refinement segment is available if and only if `Z_blk` is greater than 1, while an HT MagRef coding pass is available if and only if `Z_blk` is equal to 3.

NOTE 1 – Multiple sets of HT cleanup and HT refinement segments can be found within the codestream for a given code-block, but the decoding procedure described here processes only `Z_blk` coding passes, whose coded bytes are found within one HT cleanup segment and, if `Z_blk` is greater than 1, the one HT refinement segment that follows this HT cleanup segment.

As illustrated in Figure 1, the HT segments are comprised of byte-streams, each an ordered sequence of bytes. From each byte-stream, a bit-stream, which is an ordered sequence of bits, can be unpacked as follows:

- The magnitude and sign (MagSgn) bit-stream is recovered from the MagSgn byte-stream, which extends forward from byte 0 of the HT cleanup segment for a total of `Pcup` bytes, with prefix length, `Pcup = Lcup – Scup`; where `Lcup` is the length (in bytes) of the HT cleanup segment, and `Scup` is a suffix length.
- The adaptive run-length coding algorithm (MEL) bit-stream is recovered from the MEL byte-stream, which extends forward from byte `Pcup` of the HT cleanup segment, for at most `Scup` bytes.
- The variable length coding (VLC) bit-stream is recovered from the VLC byte-stream, which extends backward from the last byte of the HT cleanup segment, for at most `Scup` bytes. The VLC and MEL byte-streams may overlap.
- If `Z_blk` is greater than 1, the SigProp bit-stream is recovered from the SigProp byte-stream, which extends forwards from byte 0 of the HT refinement segment, for at most `Lref` bytes, where `Lref` is the length of the HT refinement segment.
- If `Z_blk` is equal to 3, the MagRef bit-stream is recovered from the MagRef byte-stream, which extends backwards from the end (byte `Lref-1`) of the HT refinement segment, for at most `Lref` bytes. The MagRef and SigProp byte-streams may overlap.

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

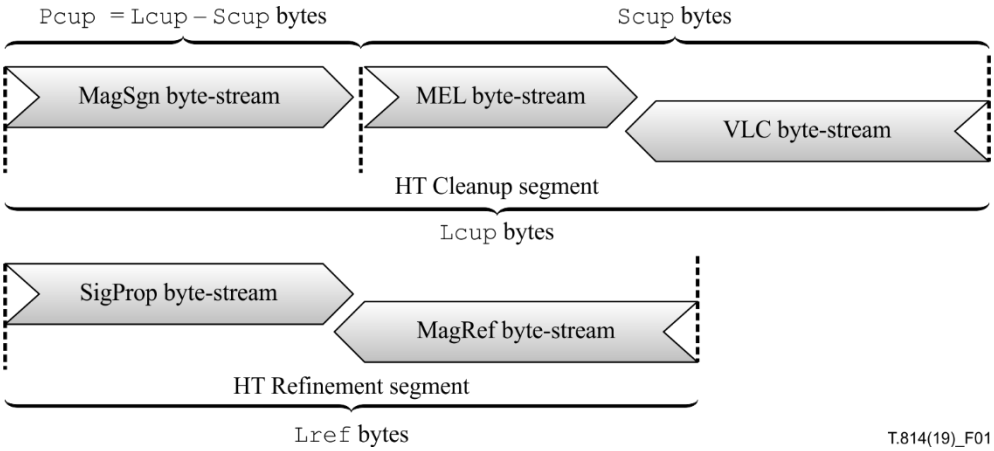


Figure 1 – HT segments and their byte-streams

The HT cleanup segment:

- shall have length `Lcup` such that $2 \leq$ `Lcup` < 65535;
- shall not contain any consecutive pair of bytes whose value, as a big-endian 16-bit unsigned integer, exceeds `0xFF8F`;
- shall not terminate with a byte whose value is `0xFF`.

The HT refinement segment:

- shall have length `Lref` satisfying $0 \leq$ `Lref` < 2047;
- shall also contain no consecutive pair of bytes whose value, as a big-endian 16-bit unsigned integer, exceeds `0xFF8F`;
- shall not terminate with a byte whose value is `0xFF`.

The suffix length `Scup` is found from the last two bytes of the HT cleanup segment as follows:

```
Scup = (16 × Dcup[Lcup-1]) + (Dcup[Lcup-2] & 0x0F)
```

where `Dcup[n]` denotes byte `n` of the HT cleanup segment, and where `n` takes value from `0` to `Lcup-1`.

After `Scup` is recovered from its last two bytes, `Dcup[n]` is accessed using the following procedure:

```
Procedure: modDcup
Returns: Modified Dcup array
State: Dcup, pos


if (pos == Lcup – 1)
  return 0xFF
else if (pos == Lcup – 2)
  return Dcup[pos] | 0x0F
else
  return Dcup[pos]
```

NOTE 2 – This procedure overwrites the last byte and the four least-significant bits (LSBs) of the second-last byte of the HT cleanup segment with 1s.

The value of `Scup` obtained in this way, shall satisfy:

$$2 \leq \texttt{Scup} \leq \min(\texttt{Lcup}, 4079)$$

Furthermore, the codestream shall be constructed such that, if `Scup` < `Lcup`, so that `Pcup` > `0`, byte `Pcup-1` of the HT cleanup segment shall not have the value `0xFF`.

NOTE 3 – The `importMagSgnBit` procedure in clause 7.1.2 effectively synthesizes a byte equal to `0xFF` to replace any byte equal to `0xFF` that might have been discarded during encoding to satisfy this constraint.

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

Details of the procedures to be used in recovering each bit-stream from its respective byte-stream are provided in clauses 7.1.2 to 7.1.6.

Similar to the HT cleanup segment, `Dref[n]` denotes byte `n` of the HT refinement segment, where `n` takes values from `0` to `Lref-1`, except that no modification is made to the bytes of the HT refinement segment.

The procedure `error()` denotes a state resulting from a codestream that does not conform to this specification, and for which behaviour is undefined.

7.1.2 Magsgn bit-stream recovery

HT MagSgn bits are retrieved from the HT MagSgn byte-stream, as required by other elements of the decoding procedure, using the `importMagSgnBit` procedure in the following. This procedure is part of a state machine with state variables `MS_pos`, `MS_bits`, `MS_tmp` and `MS_last` that are initialized using the `initMS` procedure, prior to first use of the `importMagSgnBit` procedure for an HT code-block.

```
Procedure: initMS
State: MS_pos, MS_bits, MS_tmp, MS_last


MS_pos = 0
MS_bits = 0
MS_tmp = 0
MS_last = 0
```

```
Procedure: importMagSgnBit
Returns: next MagSgn bit
State: MS_pos, MS_bits, MS_tmp, MS_last


if (MS_bits == 0)
    MS_bits = (MS_last == 0xFF)? 7 : 8
    if (MS_pos < Pcup)
        MS_tmp = modDcup(Dcup, MS_pos)
        if ((MS_tmp & (1<<MS_bits)) != 0)
            error()
    else if (MS_pos == Pcup)
        MS_tmp = 0xFF
    else
        error()
    MS_last = MS_tmp
    MS_pos = MS_pos + 1
bit = MS_tmp & 1
MS_tmp = MS_tmp >> 1
MS_bits = MS_bits - 1
return bit
```

NOTE 1 – These procedures effectively unpack bits from the HT MagSgn byte-stream in little-endian order, skipping over stuffing bits that appear in the MSB position of any byte that follows a byte equal to `0xFF`.

NOTE 2 – The value of `Pcup` can be as small as 0.

NOTE 3 – The procedure in the foregoing effectively appends at most one byte equal to `0xFF` to the HT MagSgn byte-stream, which is sufficient to allow recovery of all required HT MagSgn bits if the codestream conforms to this Specification.

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

NOTE 4 – The `importMagSgnBit` procedure is designed such that the MSB of a byte that follows a byte equal to `0xFF` is 0, unless that byte does not contribute to the MagSgn bit-stream. This is intended to simplify decoder implementations.

7.1.3 MEL bit-stream recovery

MEL bits are retrieved from the MEL byte-stream, as required by other elements of the decoding procedure, using the `importMELBit` procedure in the following. This procedure is part of a state machine with state variables `MEL_pos`, `MEL_bits`, `MEL_tmp` that are initialized using the `initMEL` procedure, prior to first use of the `importMELBit` procedure for an HT code-block.

```
Procedure: initMEL
State: MEL_pos, MEL_bits, MEL_tmp


MEL_pos = Pcup
MEL_bits = 0
MEL_tmp = 0
```

```
Procedure: importMELBit
Returns: next MEL bit
State: MEL_pos, MEL_bits, MEL_tmp


if (MEL_bits == 0)
    MEL_bits = (MEL_tmp == 0xFF) ? 7 : 8
    if (MEL_pos < Lcup)
        MEL_tmp = modDcup(Dcup,MEL_pos)
        MEL_pos = MEL_pos + 1
    else
        MEL_tmp = 0xFF
MEL_bits = MEL_bits - 1
bit = (MEL_tmp >> MEL_bits) & 1
return bit
```

NOTE – These procedures effectively unpack bits from the MEL byte-stream in big-endian order, skipping over stuffing bits that appear in the MSB position of any byte that follows a byte equal to `0xFF`.

7.1.4 HT VLC bit-stream recovery

HT VLC bits are retrieved from the HT VLC byte-stream, as required by other elements of the decoding procedure, using the `importVLCBit` procedure in the following. This procedure is part of a state machine with state variables `VLC_pos`, `VLC_bits`, `VLC_tmp` and `VLC_last` that are initialized using the `initVLC` procedure in the following, prior to first use of the `importVLCBit` procedure for an HT code-block.

```
Procedure: initVLC
State: VLC_pos, VLC_bits, VLC_tmp, VLC_last


VLC_pos = Lcup-3
VLC_last = modDcup(Dcup ,Lcup-2)
VLC_tmp = VLC_last >> 4
VLC_bits = ((VLC_tmp & 7) < 7)?4:3
```

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

```
Procedure: importVLCBit
Returns: next VLC bit
State: VLC_pos, VLC_bits, VLC_tmp, VLC_last


if (VLC_bits == 0)
    if (VLC_pos >= Pcup)
        VLC_tmp = modDcup(Dcup, VLC_pos)
    else
        error()
    VLC_bits = 8
    if (VLC_last > 0x8F) and ((VLC_tmp & 0x7F) == 0x7F)
        VLC_bits = 7
    VLC_last = VLC_tmp
    VLC_pos = VLC_pos - 1
bit = VLC_tmp & 1
VLC_tmp = VLC_tmp >> 1
VLC_bits = VLC_bits - 1
return bit
```

NOTE – These procedures effectively unpack bits from the HT VLC byte-stream in little-endian order, while consuming bytes in reverse order, skipping over stuffing bits that appear in the MSB position of any byte whose 7 LSBs are all 1s if the byte that was last consumed was larger than `0x8F`, and also skipping over the 12 bits that were replaced with 1s after using them to find the `Scup` value.

7.1.5 HT SigProp bit-stream recovery

If `Z_blk` is greater than or equal to 2, HT SigProp bits are retrieved from the HT SigProp byte-stream, as required by other elements of the decoding procedure, using the `importSigPropBit` procedure in the following. This procedure is part of a state machine with state variables `SP_pos`, `SP_bits`, `SP_tmp` and `SP_last` that are initialized using the `initSP` procedure in the following, prior to first use of the `importSigPropBit` procedure for an HT code-block.

```
Procedure: initSP
State: SP_pos, SP_bits, SP_tmp, SP_last

SP_pos = 0
SP_bits = 0

SP_tmp = 0

SP_last = 0
```

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

```
Procedure: importSigPropBit
Returns: next SigProp bit
State: SP_pos, SP_bits, SP_tmp, SP_last


if (SP_bits == 0)
    SP_bits = (SP_last == 0xFF) ? 7 : 8
    if (SP_pos < Lref)
        SP_tmp = Dref[SP_pos]
        SP_pos = SP_pos + 1
        if ((SP_tmp & (1<<SP_bits)) != 0)
            error()
    else
        SP_tmp = 0
    SP_last = SP_tmp
bit = SP_tmp & 1
SP_tmp = SP_tmp >> 1
SP_bits = SP_bits - 1
return bit
```

NOTE 1 – These procedures are similar to those used to import bits from the HT MagSgn byte-stream, except that a separate set of state variables is used, and any bytes required from beyond the `Dref` buffer are taken to be 0 – no byte equal to `0xFF` is synthesized by the decoder.

NOTE 2 – The `importSigPropBit` procedure is designed such that the MSB of a byte that follows a byte equal to `0xFF` is 0, unless that byte is not involved in the SigProp decoding process. This property can simplify decoder implementations.

7.1.6 HT MagRef bit-stream recovery

If `Z_blk` is equal to 3, HT MagRef bits are retrieved from the HT MagRef byte-stream, as required by other elements of the decoding procedure, using the `importMagRefBit` procedure in the following. This procedure is part of a state machine with state variables `MR_pos`, `MR_bits`, `MR_tmp` and `MR_last` that are initialized using the `initMR` procedure in the following, prior to first use of the `importMagRefBit` procedure for an HT code-block.

```
Procedure: initMR
State: MR_pos, MR_bits, MR_tmp, MR_last


MR_pos = Lref - 1
MR_bits = 0
MR_last = 0xFF
MR_tmp = 0
```

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

```
Procedure: importMagRefBit
Returns: next HT MagRef bit
State: MR_pos, MR_bits, MR_tmp, MR_last

if (MR_bits == 0)
    if (MR_pos >= 0)
        MR_tmp = Dref[MR_pos]
        MR_pos = MR_pos - 1
    else
        MR_tmp = 0
    MR_bits = 8
    if (MR_last > 0x8F) and ((MR_tmp & 0x7F) == 0x7F)
        MR_bits = 7
    MR_last = MR_tmp
bit = MR_tmp & 1
MR_tmp = MR_tmp >> 1
MR_bits = MR_bits - 1
return bit
```

NOTE – These procedures are similar to those used to import bits from the VLC byte-stream, except that there are no initial bits to skip and the initialization conditions are such that the MSB of the last byte in the HT MagRef byte-stream will be skipped if its seven LSBs are all 1. Also, any bytes required from before the start of the `Dref` buffer are taken to be 0.

7.2 Quad-based scanning pattern

Figure 2 illustrates the quad-based scanning pattern that is followed when decoding an HT cleanup coding pass. The HT code-block samples are arranged within an array of quads where `QW` is the width of the code-block, measured in quads, `QH` is the height of the code-block measured in quads, and

$$QW = \left\lceil \frac{Wblk}{2} \right\rceil$$

$$QH = \left\lceil \frac{Hblk}{2} \right\rceil$$

where `Wblk` and `Hblk` are the width and height of the HT code-block, measured in samples.

If `Wblk` is not divisible by 2, the HT code-block is padded with an extra column of samples on the right, so that each quad spans two sample columns. Similarly, if `Hblk` is not divisible by 2, the HT code-block is padded with an extra row of samples on the bottom, so that each quad spans two sample rows and includes exactly four samples. All padded samples shall have output values equal to 0.

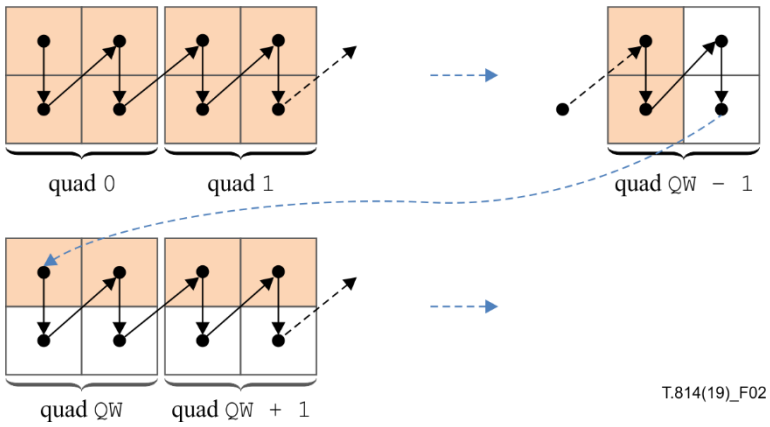


Figure 2 – Quad-based scanning pattern used in the HT cleanup pass

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

Throughout this clause, the symbol q is used to identify quads, as an index that takes values in the range

$$0 \leq q < \mathtt{QW} \times \mathtt{QH}$$

Following the quad-based scan of Figure 2, locations n within the HT code-block take values in the range

$$0 \leq n < 4 \times \mathtt{QW} \times \mathtt{QH}$$

which can also be written as

$$n = 4q + j$$

where:

$j = 0$ identifies the top-left sample within its quad;

$j = 1$ identifies the bottom-left sample within its quad;

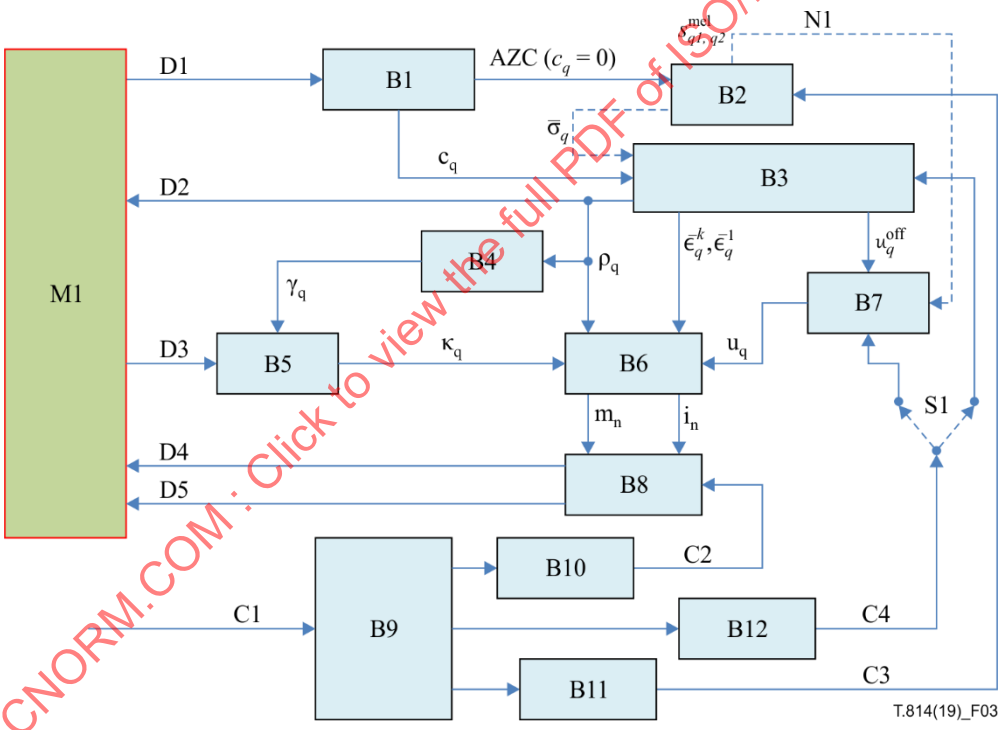
$j = 2$ identifies the top-right sample within its quad; and

$j = 3$ identifies the bottom-right sample within its quad.

7.3 HT cleanup decoding algorithm

7.3.1 Overview

Figure 3 illustrates the operation of the HT cleanup decoding algorithm.



B1: Compute contexts, as described in clause 7.3.5.
B2: Decode MEL symbols, as described in clause 7.3.3.
B3: Decode CxtVLC codewords, as described in clause 7.3.5.
B4: Compute γ_q from ρ_q, as described in clause 7.3.7.
B5: Form exponent predictors, as described in clause 7.3.7.
B6: Compute MagSgn bit counts m_n and implicit-1 flags i_n, as described in clauses 7.3.2 and 7.3.8.
B7: Decode U-VLC codewords, as described in clause 7.3.6.
B8: Decode MagSgn values, as described in clause 7.3.8.
B9: Extract byte-streams from HT cleanup segment, as described in clause 7.1.1.
B10: Extract MagSgn bit-stream from bit-stuffed MagSgn byte-stream, as described in clause 7.1.2.
B11: Extract MEL bit-stream from bit-stuffed MEL byte-stream, as described in clause 7.1.3.
B12: Extract VLC bit-stream from bit-stuffed VLC byte-stream, as described in clause 7.1.4.
C1: HT cleanup segment.
C2: MagSgn bit-stream.
C3: MEL bit-stream.
C4: VLC bit-stream.

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

D1: Retrieved neighbouring significance patterns.
D2: Generated significance patterns.
D3: Retrieved neighbouring magnitude exponents.
D4: Generated magnitude exponents.
D5: Generated code-block samples.
M1: Storage for code-block samples and derived quantities, with quad-based scanning, as described in clause 7.2.
N1: First line-pair of code-block only.
S1: De-interleave quad-pair VLC bits, as described in clause 7.3.4

Figure 3 – Operation of the HT cleanup decoding algorithm (informative). Each block in the diagram refers to the clause that defines its operation

7.3.2 Significance, exponents, predictors, MagSgn values and EMB pattern bits

This clause introduces notation and formulae that are used to describe the block-decoding procedures associated with the HT cleanup coding pass, as presented in clauses 7.3.3 to 7.3.8.

The HT cleanup coding pass produces magnitude values μ_n, along with sign values $s_n \in \{0,1\}$ for each sample of the HT code-block, where $s_n = 1$ corresponds to a negative value, and all values with zero magnitude shall have $s_n = 0$.

Sample magnitudes shall satisfy

$$0 \le \mu_n < 2^{74}$$

NOTE 1 – The upper bound here comes from the combination of a) the maximum number of bit-planes (37) for any given sub-band (see clause B.10.5 of Rec. ITU-T T.800 | ISO/IEC 15444-1), and b) the maximum value of `SPrgn` (37) allowed in HTJ2K codestreams (see clause A.5).

The significance of a sample $\sigma_n \in \{0,1\}$ identifies whether its magnitude is 0; it satisfies:

$$\sigma_n = \begin{cases} 0 & \text{if } \mu_n = 0 \\ 1 & \text{if } \mu_n > 0 \end{cases}$$

Padded samples that have been added to HT code-blocks with odd width or height, as explained in clause 7.2, shall have $\sigma_n = 0$. The significance of an entire quad q is denoted $\bar{\sigma}_q \in \{0,1\}$, indicating whether any sample in the quad is significant, and satisfies:

$$\bar{\sigma}_q = \sigma_{4q} \mid \sigma_{4q+1} \mid \sigma_{4q+2} \mid \sigma_{4q+3}$$

The significance pattern for a quad q, denoted ρ_q, is a 4-bit value comprised of the 1-bit significance values associated with each of the quad's samples; that is,

$$\rho_q = \sigma_{4q} + 2\sigma_{4q+1} + 4\sigma_{4q+2} + 8\sigma_{4q+3}$$

For significant samples ($\sigma_n = 1$) the magnitude and sign values are encapsulated within an HT MagSgn value v_n that is defined as follows:

$$v_n = 2(\mu_n - 1) + s_n$$

The magnitude exponent E_n for a sample is derived from its magnitude as follows:

$$E_n = \min\{E \in \mathbb{N} \mid (2\mu_n - 1) < 2^E\}$$

Table 1 provides a detailed elaboration of the relationship between sample magnitude μ and exponent E. No magnitude exponent shall have a value larger than 75.

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

Table 1 – Mapping of sub-band sample magnitudes to magnitude exponents

μ_p	E_p
0	0
1	1
2	2
3 to 4	3
5 to 8	4
9 to 16	5
…	…
$2^{73}+1$ to 2^{74}	75

For significant samples, the HT MagSgn value v_n is determined by unpacking m_n bits from the HT MagSgn bit-stream, as explained in clause 7.3.8 and adding $i_n \cdot 2^{m_n}$, where $i_n \in \{0,1\}$. For insignificant samples, $m_n = 0$, while for significant samples m_n is obtained by subtracting a 1-bit quantity $k_n \in \{0,1\}$ from a common exponent bound U_q for the quad q to which location n belongs. These quantities are linked by the following relationships:

$$m_n = \sigma_n \cdot U_q - k_n$$

$$i_n \le k_n \le \sigma_n$$

where the magnitude exponents of all samples in a quad q satisfy

$$E_n \le U_q \text{ for } n \in \{4q, 4q+1, 4q+2, 4q+3\}$$

The decoder determines the quad's U_q value by adding an unsigned residual value u_q to an exponent predictor κ_q.

The value of u_q is decoded in two steps, the first of which decodes an "unsigned residual offset" value $u_q^{\text{off}} \in \{0,1\}$ that indicates whether u_q is 0, while the second step decodes the value of $u_q - 1$ for quads in which u_q^{off}=1, meaning that the unsigned residual is non-zero.

The exponent max bound (EMB) pattern information for a quad q consists of two 4-bit patterns, $\bar{\epsilon}_q^{\text{k}}$ and $\bar{\epsilon}_q^{1}$, whose bits are the quantities i_n and k_n introduced in the foregoing. That is,

$$\bar{\epsilon}_q^{\text{k}} = k_{4q} + 2k_{4q+1} + 4k_{4q+2} + 8k_{4q+3}$$

and

$$\bar{\epsilon}_q^{1} = i_{4q} + 2i_{4q+1} + 4i_{4q+2} + 8i_{4q+3}$$

The significance pattern ρ_q, *EMB known bit* pattern $\bar{\epsilon}_q^{\text{k}}$ and *EMB known-1* pattern $\bar{\epsilon}_q^{1}$ are decoded together with u_q^{off}, based on a single variable length codeword for the quad q.

In this Specification, $\bar{\epsilon}_q^{\text{k}}$ and $\bar{\epsilon}_q^{1}$ are both 0 if $u_q^{\text{off}} = 0$. Moreover, if $u_q^{\text{off}} = 1$, the value of U_q shall be equal to the maximum of the magnitude exponents E_n, of the quad's samples.

NOTE 2 – The EMB patterns $\bar{\epsilon}_q^{\text{k}}$ and $\bar{\epsilon}_q^{1}$ provide information about whether individual magnitude exponents E_n are equal to the quad's maximum magnitude exponent. The variable length codewords for a quad may provide the decoder with EMB information for some, none or all samples in the quad; the *known bit* pattern $\bar{\epsilon}_q^{\text{k}}$ identifies which samples have such information. The *known-1* pattern $\bar{\epsilon}_q^{1}$ provides the EMB information itself; each bit i_n in this pattern is 1 if $k_n = 1$ and E_n is equal to the maximum exponent for the quad.

7.3.3 MEL symbol decoding procedure

The HT cleanup coding pass decoding procedure involves at most one MEL symbol s_q^{mel} for each quad q, that is retrieved by decoding the MEL bit-stream using the `decodeMELSym` procedure in the following. This procedure is part of a state machine with state variables `MEL_k`, `MEL_run` and `MEL_one` that are initialized using the `initMELDecoder` procedure in the following, prior to first use of the `decodeMELSym` procedure for an HT code-block.

The MEL decoding procedure uses an exponent table `MEL_E[]`, whose entries are listed in Table 2.

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

```
Procedure: initMELDecoder
State: MEL_k, MEL_run, MEL_one


MEL_k = 0
MEL_run = 0

MEL_one = 0
```

```
Procedure: decodeMELSym
Returns: next MEL symbol s^mel
State: MEL_k, MEL_run, MEL_one


if (MEL_run == 0) and (MEL_one == 0)
    eval = MEL_E[MEL_k]
    bit = importMELBit
    if (bit == 1)
        MEL_run = 1 << eval
        MEL_k = min(12,MEL_k+1)
    else
        MEL_run = 0
        while (eval > 0)
            bit = importMELBit
            MEL_run = 2 * MEL_run + bit
            eval = eval - 1
        MEL_k = max(0,MEL_k-1)
        MEL_one = 1
if (MEL_run > 0)
    MEL_run = MEL_run - 1
    return 0
else
    MEL_one = 0
    return 1
```

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

Table 2 – MEL Exponent Table `MEL_E[k]`

k	exponent MEL_E	k	exponent MEL_E
0	0	7	2
1	0	8	2
2	0	9	3
3	1	10	3
4	1	11	4
5	1	12	5
6	2		

© ISO/IEC 2019 – All rights reserved

7.3.4 Quad-pair interleaved decoding for the VLC bit-stream

This clause describes the quad-pair interleaving structure that shall be followed when decoding bits from the VLC bit-stream to produce significance patterns ρ_g, unsigned residuals u_g and EMB patterns $\bar{\epsilon}_q^k$ and $\bar{\epsilon}_q^1$. The decoding procedures themselves are described in clauses 7.3.5 and 7.3.6.

Figure 4 illustrates the sequence of decoding steps. For each pair of horizontally adjacent quads, the variable length decoding procedure identified as CxtVLC is performed first, for each of the quads; as described in clause 7.3.5, this consumes between 0 and 7 bits from the VLC bit-stream. Next, the variable length U-VLC prefix decoding process is performed for each of the two quads. As described in clause 7.3.6, this consumes between 0 and 3 bits from the VLC bit-stream for each of the quads, and uniquely determines the number of U-VLC suffix bits that shall occur for each of the two quads. Any U-VLC suffix bits associated with the first quad in the pair are retrieved from the VLC bit-stream before those associated with the second quad. Finally, the decoder extracts any U-VLC extension bits for the quad-pair, extracting first the extension bits for the first quad in the pair and then the extension bits for the second quad in the pair. As explained in clause 7.3.6, the number of extension bits for a quad is 0 or 4, and is determined by the value of the corresponding U-VLC suffix.

U-VLC extensions shall have zero length in cases where the sample magnitudes μ_n cannot exceed 2^{36}.

EXAMPLE – If the parameter *B* specified in clause 8.7.3 is less than or equal to 36, a decoder can rely upon the fact that there will be no U-VLC extensions.

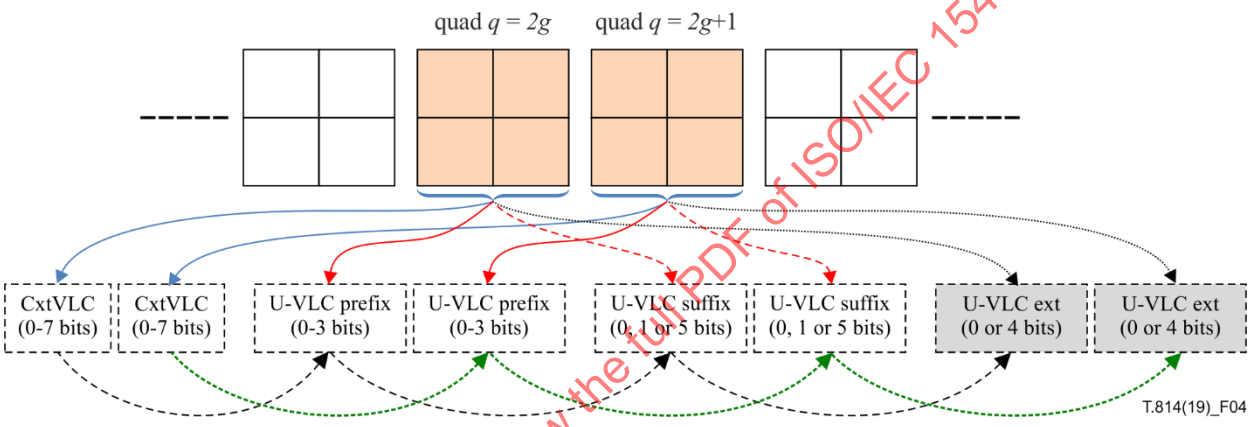


Figure 4 – Quad-pair interleaving of VLC decoding steps
(Arrows identify dependencies)

If QW is odd, the final quad-pair on each row only has a first quad. For such quad-pairs, the decoder shall not perform any of the decoding steps suggested in the foregoing for the missing second quad.

7.3.5 Decoding of significance and EMB patterns and unsigned residual offsets

This clause describes the context-adaptive variable length decoding procedure that is used to decode the significance pattern ρ_q, the unsigned residual offset u_q^{off} and the EMB patterns $\bar{\epsilon}_q^k$ and $\bar{\epsilon}_q^1$ for each quad q.

The decoding procedure depends upon a context value c_q that is computed from the significance of a set of neighbouring samples, as shown in Figure 5.

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

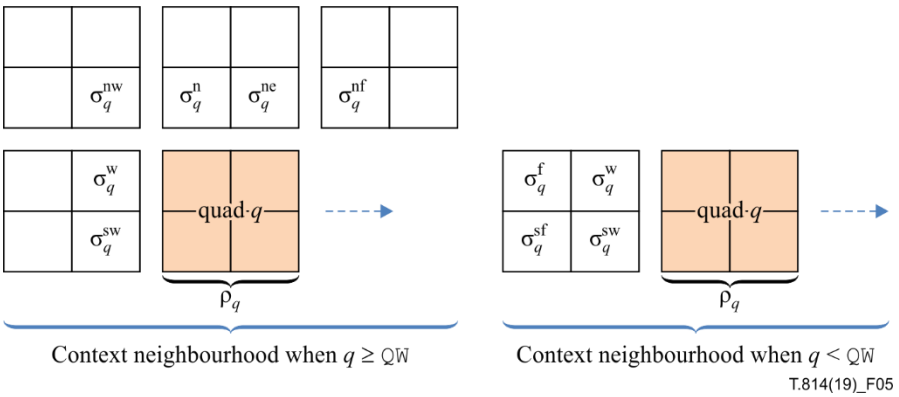


Figure 5 – Significance neighbourhood information used to form coding contexts c_q for quads found in non-initial ($q \geq$ QW) and initial ($q <$ QW) line-pairs within a HT code-block

The superscript labels (nw, n, ne, nf, w, sw, f, and sf) are used to identify the relevant neighbours

The neighbours used in the first row of quads for the HT code-block, where $q <$ QW, are denoted

$$(\sigma_q^{sw}, \sigma_q^{w}, \sigma_q^{sf}, \sigma_q^{f}) = \begin{cases} (\sigma_{4q-1}, \sigma_{4q-2}, \sigma_{4q-3}, \sigma_{4q-4}) & \text{if } q > 0 \\ (0,0,0,0) & \text{if } q = 0 \end{cases}$$

and in this case the context value is computed as follows:

$$c_q = (\sigma_q^{f} | \sigma_q^{sf}) + 2\sigma_q^{w} + 4\sigma_q^{sw} \quad (1)$$

The neighbours used in non-initial quad rows of the HT code-block, where $q \geq$ QW, are denoted

$$\sigma_q^{n} = \sigma_{4(q-QW)+1}, \qquad \sigma_q^{ne} = \sigma_{4(q-QW)+3},$$

$$\sigma_q^{nw} = \begin{cases} \sigma_{4(q-QW)-1} & \text{if } \mathrm{mod}(q,QW) \neq 0 \\ 0 & \text{otherwise} \end{cases} \text{ and } \sigma_q^{nf} = \begin{cases} \sigma_{4(q-QW)+5} & \text{if } \mathrm{mod}(q+1,QW) \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

and in this case the context value is computed as follows:

$$c_q = (\sigma_q^{nw} | \sigma_q^{n}) + 2(\sigma_q^{w} | \sigma_q^{sw}) + 4(\sigma_q^{ne} | \sigma_q^{nf}) \quad (2)$$

Quads q for which $c_q = 0$ are identified as all zero context (AZC) quads and receive special treatment in the decoding process, which is represented by the `decodeSigEMB` procedure in the following. This procedure relies upon the `decodeMELSym` procedure, as well as a `decodeCxtVLC` procedure in the following, which is used for non-AZC quads and for AZC quads that are determined to be significant because the `decodeMELSym` procedure returns a 1.

The `decodeCxtVLC` procedure itself is based on a separate prefix code for each context c_q. Prefix codes are further differentiated based on quad-type – i.e., whether the quad being decoded belongs to the first row of quads for the HT code-block ($q <$ QW) or not. Bits from the VLC bit-stream are imported using the `importVLCbit` procedure described in clause 7.1, until one of the codewords of the prefix code is matched, revealing the tuple $(\rho_q, u_q^{off}, \bar{\epsilon}_q^{k}, \bar{\epsilon}_q^{1})$ as the decoded result. Annex C provides the code tables `CxtVLC_table_0` and `CxtVLC_table_1`, corresponding to each quad-type. The function `test_match` returns true if a codeword prefix `cwd`, length `len` and context c_q match the w, l_w and c_q fields of an entry in the table. The function `get_match` returns the $(\rho_q, u_q^{off}, \bar{\epsilon}_q^{k}, \bar{\epsilon}_q^{1})$ values from the matching entry.

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

```
Procedure: decodeCxtVLC
Input: quad index q and context c_q
Returns: (ρ_q, u_q^off, ε̄_q^k, ε̄_q^1) tuple for quad q

if (q < QW)
    table = CxtVLC_table_0
else
    table = CxtVLC_table_1
len = 1
cwd = importVLCbit
while (!test_match(table, c_q, cwd, len))
    bit = importVLCbit
    cwd = cwd | (bit << len)
    len = len + 1
(ρ_q, u_q^off, ε̄_q^k, ε̄_q^1) = get_match(table, c_q, cwd, len)
return (ρ_q, u_q^off, ε̄_q^k, ε̄_q^1)
```

```
Procedure: decodeSigEMB
Input: quad index q and context c_q
Returns: (ρ_q, u_q^off, ε̄_q^k, ε̄_q^1) tuple for quad q

if (c_q == 0)
    sym = decodeMELSym
    if (sym == 0)
        return (ρ_q, u_q^off, ε̄_q^k, ε̄_q^1) = (0,0,0,0)
(ρ_q, u_q^off, ε̄_q^k, ε̄_q^1) = decodeCxtVLC
return (ρ_q, u_q^off, ε̄_q^k, ε̄_q^1)
```

7.3.6 Decoding of unsigned residuals

This clause describes the procedure used to decode the unsigned residual value u_q for a quad q in which the decoded value of u_q^{off} is 1. If $u_q^{\text{off}} = 0$, the unsigned residual u_q is 0 and no further unsigned residual decoding is required for the quad.

When $u_q^{\text{off}} = 1$, the value of u_q is decoded with the aid of a U-VLC variable length decoding procedure that involves up to three steps. The first step is to decode the variable length U-VLC prefix. For certain prefix values, a second step is required, in which a U-VLC suffix is decoded. The number of U-VLC suffix bits that need to be decoded from the VLC bit-stream is determined entirely by the U-VLC prefix value. If the U-VLC suffix value is greater than 27, a third step is required, in which 4 bits are imported from the VLC bit-stream to form a U-VLC extension code. The U-VLC prefix, suffix and extension decoding steps for each pair of quads are interleaved, as described in clause 7.3.4.

Table 3 provides the complete U-VLC code that is used as-is to decode the u_q value for quads belonging to a non-initial line-pair of the HT code-block – i.e., whenever $q \geq$ QW. The same method is used to decode the u_q values for quad-pairs belonging to the initial line-pair of the HT code-block, where quads q_1 and q_2 of the quad-pair do not both have $u_{q_1}^{\text{off}} = 1$ and $u_{q_2}^{\text{off}} = 1$. The final decoded value for u_q in these cases is:

$$u = \text{u_pfx} + \text{u_sfx} + 4 * \text{u_ext} \tag{3}$$

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

where `u_pfx` is decoded using the `decodeUPrefix` procedure, then `u_sfx` is decoded using the `decodeUSuffix` procedure, then `u_ext` is decoded using the `decodeUExtension` procedure, all of which appear in the following.

Table 3 – U-VLC code used to encode unsigned residuals $u > 0$. The prefix string here is matched against bits from the VLC bit-stream from left to right, consuming $l_p(u)$ bits. The suffix and extension words are unsigned integers with $l_s(u)$ and $l_e(u)$ bits that are imported from the VLC bit-stream in little-endian order (i.e., least significant bit first)

u	Prefix		Suffix	Extension	$l_p(u)$	$l_s(u)$	$l_e(u)$	$l_p(u) + l_s(u) + l_e(u)$
	cwd	u_pfx	u_sfx	u_ext				
1	"1"	1	--	--	1	0	0	1
2	"01"	2	--	--	2	0	0	2
3	"001"	3	$(u-3)$	--	3	1	0	4
4	"001"	3	$(u-3)$	--	3	1	0	4
5	"000"	5	$(u-5)$	--	3	5	0	8
6	"000"	5	$(u-5)$	--	3	5	0	8
…	…	…	…	--	…	…	…	…
32	"000"	5	27	--	3	5	0	8
33	"000"	5	28+mod((u-33),4)	$\lfloor (u-33)/4 \rfloor$	3	5	4	12
34	"000"	5	28+mod((u-33),4)	$\lfloor (u-33)/4 \rfloor$	3	5	4	12
…	…	…	…	…	…	…	…	12
74	"000"	5	29	10	3	5	4	12

```
Procedure: decodeUPrefix
Returns: U-VLC prefix value u_pfx


bit = importVLCBit
if (bit == 1) return 1
bit = importVLCBit
if (bit == 1) return 2
bit = importVLCBit
return (bit == 1)? 3:5
```

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

```
Procedure: decodeUSuffix
Input: U-VLC prefix value u_pfx
Returns: U-VLC suffix value u_sfx


if (u_pfx < 3) return 0
val = importVLCBit
if (u_pfx == 3) return val
for (i=1; i < 5; i++)
    bit = importVLCBit
    val = val + (bit << i)
return val
```

```
Procedure: decodeUExtension
Input: U-VLC suffix value u_sfx
Returns: U-VLC extension value u_ext


if (u_sfx < 28) return 0
val = importVLCBit
for (i=1; i < 4; i++)
    bit = importVLCBit
    val = val + (bit << i)
return val
```

For quads belonging to the first line-pair of the HT code-block (i.e., when $q < \mathtt{QW}$), the process used to decode a non-zero u_q value (i.e., when u_q^{off} is 1) is modified where quads q_1 and q_2 of a quad-pair are found to have both $u_{q_1}^{\text{off}} = 1$ and $u_{q_2}^{\text{off}} = 1$. In this case, a single MEL symbol $s_{q_1q_2}^{\text{mel}}$ is decoded for the quad-pair by invoking the `decodeMELSym` procedure. For clarity, this is done after the `decodeCxtVLC` steps have been performed for the quad-pair to which quad q belongs. The decoding of u_{q_1} and u_{q_2} then proceeds in one of two ways, depending upon the value of the decoded MEL symbol $s_{q_1q_2}^{\text{mel}}$. If $s_{q_1q_2}^{\text{mel}} = 1$, the `decodeUPrefix`, `decodeUSuffix` and `decodeUExtension` procedures are used to decode prefix, suffix and extension values `u_pfx`, `u_sfx` and `u_ext`, exactly as in the foregoing, for each quad, and the decoded u_q values are found from

$$u = 2 + \mathtt{u_pfx} + \mathtt{u_sfx} + 4 * \mathtt{u_ext} \tag{4}$$

Otherwise, if $s_{q_1q_2}^{\text{mel}} = 0$, the first quad's unsigned residual u_{q_1} is found using Formula (3), but decoding of the second quad's unsigned residual u_{q_2} depends upon the decoded u_{q_1} values. Specifically, where $u_{q_1} > 2$, the U-VLC prefix decoding step for u_{q_2} is replaced by using `importVLCBit` directly to import a single bit u_{bit} from the VLC bit-stream and setting `u_pfx` $= u_{\text{bit}} + 1$; the decoded u_{q_2} value is then

$$u_{q_2} = u_{\text{bit}} + 1$$

Where $u_{q_1} \leq 2$ the decoding of u_{q_2} proceeds in the same way as u_{q_1}, using Formula (3).

NOTE – When $u_{q_1}^{\text{off}} = u_{q_2}^{\text{off}} = 1$ and $s_{q_1q_2}^{\text{mel}} = 0$, the condition $u_{q_1} > 2$ means that the `decodeUPrefix` procedure for the first quad returns `u_pfx` > 2, or equivalently, that the first quad's U-VLC prefix has length 3.

7.3.7 Determination of predictors and exponent bounds

This clause describes the procedure by which a decoder computes exponent predictors κ_q for each quad q, and combines these with the unsigned residual values u_q to deduce exponent bounds U_q.

For the first row of quads in an HT code-block ($q < \mathtt{QW}$), the exponent predictor satisfies

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

$$\kappa_q = 1$$

For all other quads, κ_q is computed from the magnitude exponents of neighbouring decoded samples from the preceding line in the HT code-block, as illustrated in Figure 6. Specifically, the exponents that are used are:

$$E_q^{\mathrm{n}} = E_{4(q-\mathrm{QW})+1}, \qquad E_q^{\mathrm{ne}} = E_{4(q-\mathrm{QW})+3},$$

$$E_q^{\mathrm{nw}} = \begin{cases} E_{4(q-\mathrm{QW})-1} & \text{if } \mathrm{mod}(q,\mathrm{QW}) \neq 0 \\ 0 & \text{otherwise} \end{cases} \quad \text{and} \quad E_q^{\mathrm{nf}} = \begin{cases} E_{4(q-\mathrm{QW})+5} & \text{if } \mathrm{mod}(q+1,\mathrm{QW}) \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

The decoder derives these exponents from decoded sample magnitudes μ_n, using the procedure expounded via Table 1.

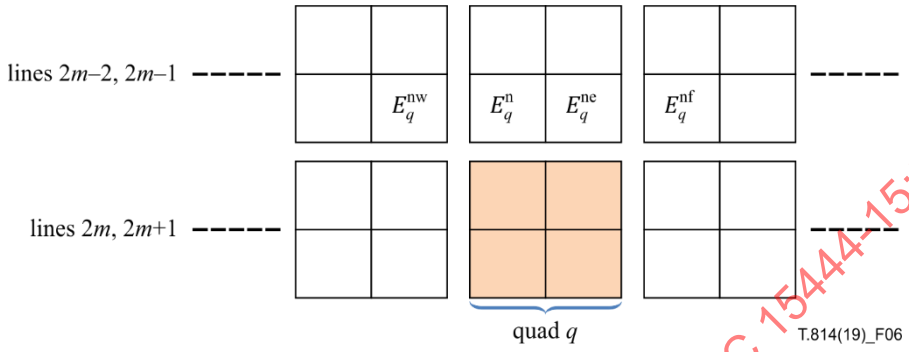


Figure 6 – Neighbourhood information used to form exponent predictors for quads in non-initial line-pairs of a block

These exponents are converted to an exponent predictor κ_q, using

$$\kappa_q = \max\{1, \gamma_q \cdot (\max\{E_q^{\mathrm{nw}}, E_q^{\mathrm{n}}, E_q^{\mathrm{ne}}, E_q^{\mathrm{nf}}\} - 1)\}, \tag{5}$$

where $\gamma_q \in \{0,1\}$ indicates whether quad q has more than one significant sample. Specifically,

$$\gamma_q = \begin{cases} 0 & \text{if } \rho_q \in \{0,1,2,4,8\} \\ 1 & \text{otherwise} \end{cases} \tag{6}$$

The exponent bound U_q for quad q is obtained from

$$U_q = \kappa_q + u_q$$

The decoded unsigned residual u_q shall have the smallest non-negative value that is consistent with the constraint

$$U_q \geq E_n \text{ for each } n \in \{4q, 4q+1, 4q+2, 4q+3\}$$

where E_n is the magnitude exponent associated with each decoded sample magnitude μ_n.

7.3.8 Unpacking the HT MagSgn bit-stream

Using the exponent bound U_q, significance pattern ρ_q and EMB patterns $\bar{\epsilon}_q^{\mathrm{k}}$ and $\bar{\epsilon}_q^{1}$ for each quad, the decoder determines the number of bits m_n according to

$$m_n = \sigma_n \cdot U_q - k_n \text{ for each } n \in \{4q, 4q+1, 4q+2, 4q+3\}$$

and then recovers the HT MagSgn values for each sample, following the scanning pattern in Figure 2, by using procedure `decodeMagSgnValue` that appears in the following. Here, σ_n, k_n and i_n are the individual bits of the 4-bit patterns ρ_q, $\bar{\epsilon}_q^{\mathrm{k}}$ and $\bar{\epsilon}_q^{1}$, respectively, as explained in clause 7.3.2.

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

```
Procedure: decodeMagSgnValue
Input: number of mag-sign bits m_n and known-1 value i_n for a sample n
Returns: HT MagSgn value v_n for the sample


val = 0
for (i=0; i < m_n; i++)
    bit = importMagSgnBit
    val = val + (bit << i)
val = val + (i_n << m_n)
return val
```

From the decoded HT MagSgn values v_n, the decoder recovers magnitude values μ_n and sign bits s_n as follows:

$$\mu_n = \begin{cases} \lfloor v_n/2 \rfloor + 1 & \text{if } m_n \neq 0 \\ 0 & \text{if } m_n = 0 \end{cases} \quad s_n = \begin{cases} \text{mod}(v_n, 2) & \text{if } m_n \neq 0 \\ 0 & \text{if } m_n = 0 \end{cases}$$

NOTE – Before the `decodeMagSgnValue` procedure can be used to reconstruct HT MagSgn values within a current non-initial row of quads, samples from the preceding row of quads must be decoded and at least some of them converted to magnitude exponents, so as to enable computation of κ_q and U_q values for the current non-initial row of quads.

7.4 HT SigProp decoding procedure

This clause describes the procedure for decoding an HT SigProp coding pass, which is performed when `Z_blk` is greater than 1. The decoder uses significance information σ_n produced by decoding the HT cleanup pass, together with the HT SigProp bit-stream, to recover binary refinement values $r_n \in \{0,1\}$ and refinement indicators $z_n \in \{0,1\}$ for each sample in the HT code-block. Prior to performing the HT SigProp decoding procedure, the r_n and z_n values for all samples in the HT code-block are set to 0. The decoder then progressively updates these values, depending on the significance information from the HT cleanup pass, as well as the bits found within the HT SigProp bit-stream. During this process, additional sign values s_n are also decoded for samples where $r_n = 1$.

HT SigProp decoding follows the same four-line stripe-oriented scanning pattern as the block decoder defined in REC. ITU-T T.800 | ISO/IEC 15444-1, which is illustrated in Figure 7. In this Specification, however, the location n that is used to identify individual samples conforms to the notation introduced in clause 7.2, corresponding to the quad-based scanning order. In following the stripe-oriented scan of Figure 7, the decoder shall skip any location that lies outside the HT code-block, which means that the last stripe in the block is truncated, if necessary, to $\texttt{Hblk} - 4 \cdot \lfloor (\texttt{Hblk} - 1)/4 \rfloor$ lines.

To facilitate the explanation, two neighbourhoods of the sample at location n are introduced: a propagation neighbourhood $\mathcal{N}_n$; and a scan-causal neighbourhood $\bar{\mathcal{N}}_n$.

If bit 3 of the SPcod or SPcoc field is 0 (see clause A.4), the propagation neighbourhood $\mathcal{N}_n$ for a sample consists of all locations within the HT code-block that are immediate neighbours of the sample with location n; for clarity, there are eight such neighbours for all samples apart from those that lie on the boundaries of the HT code-block.

If bit 3 of the SPcod or SPcoc field is 1 (see clause A.4), the propagation neighbourhood $\mathcal{N}_n$ for a sample consists of all locations within the same stripe or a previous stripe, that are immediate neighbours of the sample with location n.

NOTE 1 – Samples on the last line of a stripe have at most six propagation neighbours in this case, while samples on other lines within a stripe have at most eight propagation neighbours.

The scan-causal neighbourhood $\bar{\mathcal{N}}_n$ is the subset of $\mathcal{N}_n$ corresponding to samples that appear earlier than the sample with location n in the stripe-oriented scan.

NOTE 2 – As illustrated in Figure 7, the location of a sample in the stripe-oriented scan affects the number of samples that belong to its scan-causal neighbourhood.

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

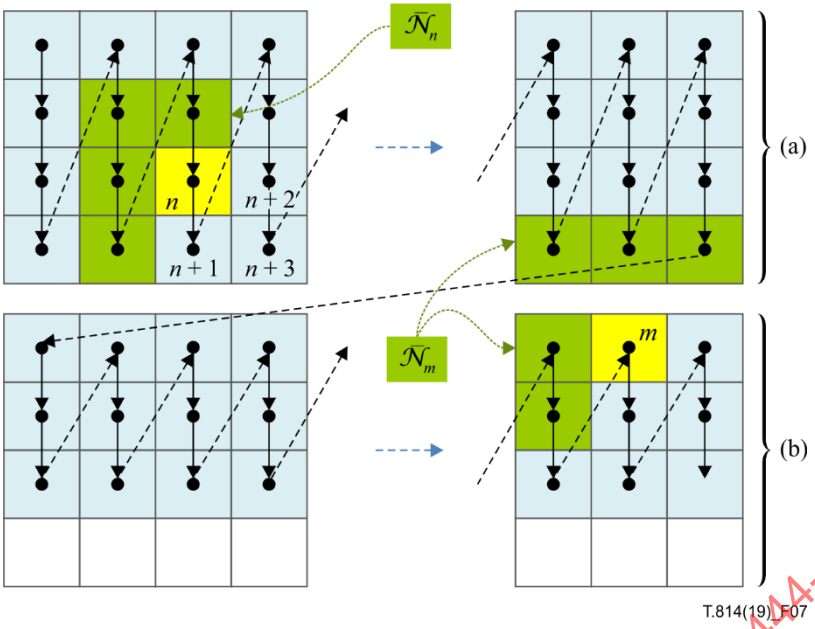


Figure 7 – Stripe-oriented scan illustrating scan-causal neighbourhoods $\bar{\mathcal{N}}_n$ for a sample n, where sample indices are ordered according to the quad-based scanning convention of Figure 2. In the example, the HT code-block has height H=7, so the last stripe (b) in the stripe-oriented scan has only three lines, whereas stripe (a) has four

The HT SigProp decoding procedure involves a magnitude decoding step and a sign decoding step, that are interleaved on a quad-column basis. Following the stripe-oriented scan of Figure 7, the decoder performs the magnitude decoding step by invoking the `decodeSigPropMag` procedure in the following, for four stripe columns (a column-group), after which it passes through the same samples a second time, in the same order, performing the sign decoding step using the `decodeSigPropSign` procedure in the following. The process is repeated for all column-groups in a stripe, before proceeding with the next stripe. If the HT code-block width `Wblk` is not divisible by 4, the number of columns in the final column-group of each stripe is reduced to `mod(Wblk,4)`.

```
Procedure: decodeSigPropMag
Input: sample location n, significance values σ and existing refinement values r
Side effects: May change z_n and r_n


mbr = 0
if (σ_n == 0)
    for each m ∈ N̄_n
        mbr = mbr | σ_m
    for each m ∈ N̄_n
        mbr = mbr | r_m
if (mbr != 0)
    set z_n = 1
    set r_n = importSigPropBit
```

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

```
Procedure: decodeSigPropSign
Input: sample location n and refinement value r_n
Side effects: May change sign value s_n


if (r_n != 0)
    set s_n = importSigPropBit
```

7.5 HT MagRef decoding procedure

This clause describes the procedure for decoding an HT MagRef coding pass, which is performed when `Z_blk` is equal to 3. The decoder uses significance information σ_n produced by decoding the HT cleanup pass, together with the HT MagRef bit-stream, to recover binary refinement values $r_n \in \{0,1\}$ and refinement indicators $z_n \in \{0,1\}$ for each sample in the HT code-block. Prior to performing the HT MagRef decoding procedure, r_n and z_n for each sample have the values determined by the HT SigProp decoding procedure, which shall be performed first.

HT MagRef decoding follows the same four-line stripe-oriented scanning pattern as the HT SigProp coding pass, as illustrated in Figure 7. Again, however, the location n that is used to identify individual samples here conforms to the notation introduced in clause 7.2, corresponding to the quad-based scanning order.

The HT MagRef decoding procedure involves only a magnitude decoding step that is performed by applying the `decodeMagRefValue` procedure in the following to each sample, following the four-line stripe-oriented scan.

```
Procedure: decodeMagRefValue
Input: sample location n and significance value σ_n
Side effects: May change z_n and r_n


if (σ_n != 0)
    set z_n = 1
    set r_n = importMagRefBit
```

7.6 Sample output values

This clause describes the process whereby a decoder converts decoded magnitude values μ_n, sign values s_n, refinement values r_n and refinement indicators z_n into values for processing by the inverse quantization procedure defined in Annex E of Rec. ITU-T T.800 | ISO/IEC 15444-1, after the application of any applicable region of interest transformation, as described in Annex H of Rec. ITU-T T.800 | ISO/IEC 15444-1. Following the notation in Rec. ITU-T T.800 | ISO/IEC 15444-1, these values are the number of magnitude bits $N_b(x,y)$, the magnitude bits $\mathtt{MSB}_i(b,x,y)$ and the sign bits $s_b(x,y)$, where (x,y) identifies a location within sub-band b, and $1 \le i \le N_i(x,y)$.

The procedure here depends upon the quantity `S_blk` that identifies the number of skipped magnitude bit-planes associated with the HT cleanup coding pass, as described in Annex B. In what follows, b identifies the sub-band to which a decoded HT code-block belongs, and (x_n, y_n) denotes the sub-band-based coordinates of the sample with location n in this decoded HT code-block.

The number of decoded magnitude bit-planes is found from

$$N_b(x_n, y_n) = \mathtt{S_blk} + 1 + z_n$$

while the sign bits are assigned as

$$s_b(x_n, y_n) = s_n$$

For each i in the range $1 \le i \le \mathtt{S_blk} + 1$, the magnitude bit $\mathtt{MSB}_i(b,x,y)$ is given by

$$\mathtt{MSB}_i(b, x_n, y_n) = \mathrm{mod}\left(\left\lfloor \frac{\mu_n}{2^{\mathtt{S_blk}+1-i}} \right\rfloor, 2\right)$$

Finally, if z_n is non-zero,

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

$$\mathrm{MSB}_{\mathrm{S_blk}+2}(b, x_n, y_n) = r_n$$

8 Constrained codestream sets

8.1 Overview

Clause 8 defines sets of HTJ2K codestreams, each conforming to one or more specified constraints.

These sets partition the space of all possible HTJ2K codestreams as a function of implementation throughput and complexity. They are provided to simplify the task of creating profiles and defining decoder capabilities.

EXAMPLE – In order to reduce implementation complexity, a profile definition can specify that only HTJ2K codestreams that belong to the HTONLY set specified in clause 8.2 are permitted.

8.2 HTONLY, HTDECLARED and MIXED sets

The HTONLY set is the set of HTJ2K codestreams where all code-blocks are HT code-blocks.

The HTDECLARED set is the set of HTJ2K codestreams where all code-blocks within a given tile-component are either a) HT code-blocks, or b) code-blocks as specified in Rec. ITU-T T.800 | ISO/IEC 15444-1.

The MIXED set is the set of all HTJ2K codestreams that are not in the HTDECLARED set.

NOTE 1 – A codestream that belongs to the HTONLY set also belongs to the HTDECLARED set, but the converse is not true. In particular, an HTJ2K codestream where all code-blocks conform to Rec. ITU-T T.800 | ISO/IEC 15444-1 belong to the HTDECLARED set, but not the HTONLY set.

NOTE 2 – Decoding of HTJ2K codestreams that belong to the HTONLY set does not require decoding of code-blocks that conform to Rec. ITU-T T.800 | ISO/IEC 15444-1.

8.3 SINGLEHT and MULTIHT sets

The SINGLEHT set is the set of HTJ2K codestreams where at most one HT set is ever present for each HT code-block.

The MULTIHT set is the set of all HTJ2K codestreams that are not in the SINGLEHT set.

NOTE – Although a decoder is not required to decode more than one HT set for any HT code-block, parsing is more complex for codestreams in the MULTIHT set.

EXAMPLE – The use of multiple non-empty HT sets in a code-block results in redundancy. This can be used, for instance, in content distribution systems to avoid the need for decoding and re-encoding code-blocks when transcoding to different coded data rates. The resulting transcoded codestream, however, normally contains at most one HT set per HT code-block.

8.4 RGN and RGNFREE sets

The RGNFREE set is the set of HTJ2K codestreams that do not contain any RGN marker segment.

The RGN set is the set of all HTJ2K codestreams that are not in the RGNFREE set.

NOTE 1 – Whether an HTJ2K codestream contains RGN marker segments impacts HT cleanup magnitude bounds as described in clause 8.7.3.

NOTE 2 – RGN marker segments are typically used in the progressive communication of images that contain defined spatial regions of interest; however, the HT block-coder does not provide an effective mechanism for progressive coding.

NOTE 3 – The presence of RGN marker segments complicates the dequantization procedure that is applied after block decoding.

8.5 HOMOGENEOUS and HETEROGENEOUS sets

The HOMOGENEOUS set is the set of HTJ2K codestreams where:

- none of the functional marker segments, e.g., COD, COC, RGN, QCD, QCC, and POC, are present in any tile-part header; and
- no PPT marker segment is present.

The HETEROGENEOUS set is the set of all HTJ2K codestreams that are not in the HOMOGENEOUS set.

NOTE – Decoder configuration information can be retrieved entirely from the main header if the HTJ2K codestream belongs to the HOMOGENEOUS set. Conversely, decoding codestreams that belong to the HETEROGENEOUS set can require a decoder to be reconfigured between tiles, which cannot be done until after its first tile-part is encountered.

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

8.6 LOCAL and FRAG sets

The LOCAL set is the set of HTJ2K codestreams where:

- the exponents PPx and PPy satisfy PPx + PPy ≤ 16, except in the lowest resolution level within each tile-component, where PPx + PPy ≤ 14; and
- either (i) the codestream has only one quality layer, as identified via the SGcod parameter, or (ii) the progression order value for the SGcod, SPcoc, and Ppoc parameters is in the range 2 to 4 (see Table A.16 at Rec. ITU-T T.800 | ISO/IEC 15444-1).

The FRAG set is the set of all HTJ2K codestreams that are not in the LOCAL set.

NOTE 1 – The first condition in the foregoing limits the extent to which co-located code-blocks from sub-bands with different orientations can be separated within the codestream. The second condition prevents individual code-block byte-streams from being fragmented across non-consecutive packets within the codestream. Together, these conditions can reduce the amount of compressed data re-ordering needed when decoding a codestream.

NOTE 2 – Fragmentation is typically used in combination with quality scalability; however, the HT block-coder does not provide an effective mechanism for progressive coding.

8.7 Bounded magnitude sets

8.7.1 Overview

Clause 8.7 defines sets that correspond to bounds on the magnitudes μ_n that are produced by the HT cleanup decoding procedure. The bounds depend upon whether an irreversible spatial wavelet transformation is employed, so there are two types of bounded magnitude sets: those in which an irreversible transform is associated with HT code-blocks; and those where only reversible transforms are associated with HT code-blocks.

NOTE – Unlike the block-decoding algorithm specified in Rec. ITU-T T.800 | ISO/IEC 15444-1, the HT block-coding algorithm does not allow decoders to discard bits in order to accommodate limitations to their internal working precision. As a result, the magnitude bound affects implementation complexity. At the time of this writing, a reasonable magnitude bound for CPU-based implementations is $B = 31$, where the interpretation of B is found in clause 8.7.3.

8.7.2 HTIRV and HTREV sets

The high-throughput reversible (HTREV) set is the set of HTJ2K codestreams where every tile-component that contains one or more HT code-blocks signals a reversible transform.

The high-throughput irreversible (HTIRV) set is the set of all HTJ2K codestreams that are not in the HTREV set.

NOTE 1 – The use of reversible transforms impacts the MAGB$_P$ sets, as described in clause 8.7.3.

NOTE 2 – In transcoding operations that reduce image resolution by discarding the k highest resolution levels from each tile-component, the magnitude bound B is increased by k for codestreams in the HTIRV set, while no such adjustment is made for codestreams in the HTREV set.

8.7.3 MAGB$_P$ sets

Each MAGB$_P$ set specified in Table 4 is associated with a value of the parameter B, and consists of the HTJ2K codestreams where all HT Cleanup magnitudes μ_n of a given sub-band b are smaller than μ_{bound}, where:

$\mu_{bound} = 2^B$, if $B > 31$ or the sub-band's transformation type is not an irreversible transform; or

$\mu_{bound} = 2^{\min\{31, B+\lceil n_b \rceil - 1\}}$ where n_b is the sub-band decomposition level, otherwise.

NOTE 1 – If no RGN marker segment is present in the codestream, then all magnitudes μ_n necessarily satisfy $\mu_n < 2^{37}$. If RGN marker segments are present in an HTJ2K codestream, then all magnitudes M necessarily satisfy $\mu_n < 2^{74}$.

NOTE 2 – If the arbitrary decomposition extensions specified in Annex F of Rec. ITU-T T.801 | ISO/IEC 15444-2 is used, the sub-band decomposition level n_b can be an even or an odd integer multiple of ½. The expression $\lceil n_b \rceil$ ensures that μ_{bound} is always an integer power of 2.

EXAMPLE – A decoder can increase throughput by using a hardware-accelerated implementation if the HT cleanup magnitudes are below a given threshold, i.e., if the HTJ2K codestream belongs to a set where parameter B is below a certain threshold; and reverting to a slower software implementation otherwise.

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

Table 4 – HT cleanup magnitudes bound codestream sets

Set MAGBP	Parameter B
$MAGB_0$	8
$MAGB_1$	9
$MAGB_2$	10
$MAGB_3$	11
$MAGB_4$	12
$MAGB_5$	13
$MAGB_6$	14
$MAGB_7$	15
$MAGB_8$	16
$MAGB_9$	17
$MAGB_{10}$	18
$MAGB_{11}$	19
$MAGB_{12}$	20
$MAGB_{13}$	21
$MAGB_{14}$	22
$MAGB_{15}$	23
$MAGB_{16}$	24
$MAGB_{17}$	25
$MAGB_{18}$	26
$MAGB_{19}$	27
$MAGB_{20}$	31
$MAGB_{21}$	35
$MAGB_{22}$	39
$MAGB_{23}$	43
$MAGB_{24}$	47
$MAGB_{25}$	51
$MAGB_{26}$	55
$MAGB_{27}$	59
$MAGB_{28}$	63
$MAGB_{29}$	67
$MAGB_{30}$	71
$MAGB_{31}$	74

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

8.8 CPF_N sets

The set CPF_N consists of all HTJ2K codestreams that can be obtained using the following procedure.

- Let C1 be an Rec. ITU-T T.800 | ISO/IEC 15444-1 codestream conforming to profile *N*.
- Let C2 be an HTJ2K codestream.
- Block transcoding. Each code-block in C2 is either unchanged from C1 or transcoded to an HT code-block by decoding and then re-encoding such that it conforms to Annex B. The final HT cleanup pass of each transcoded code-block corresponds to the final cleanup pass from the original code-block in C1, except where this cleanup pass involve sample magnitudes μ_n that are inconsistent with constraints imposed on C2, in which case the smallest number of original coding passes necessary to avoid such inconsistency are discarded. If, after any such discarding, the original code-block in C1 has a SigProp pass that follows the final cleanup pass, the transcoded code-block in C2 has a corresponding HT SigProp pass. Similarly, if the original code-block in C1 has a MagRef pass that follows the final non-discarded cleanup pass, the transcoded code-block in C2 has a corresponding HT MagRef pass.

NOTE 1 – Constraints on C2 are signalled using the CAP marker segment, as specified in clause A.3, and the PRF marker segment, as specified in Rec. ITU-T T.800 | ISO/IEC 15444-1.

© ISO/IEC 2019 – All rights reserved

- Packetization. For each packet of C1, there shall be a corresponding packet of C2 and vice-versa. The code-block-coding passes within each packet in C2 shall be identical to the coding passes included within each packet in C1, with the exception only of those coding passes from C1 that are discarded as specified in the foregoing.
- Marker segment generation. All functional, fixed information and pointer marker segments from C1 are preserved in C2, except that the values of the SPcod/SPcoc parameter in COD/COC marker segments are modified to reflect the use of HT block coding, the SIZ, CAP and PRF marker segments are modified or introduced according to the requirements of Annex A, and SOT, PLT, PLM and TLM marker segments are updated to reflect the lengths of the transcoded code-blocks. No new marker segments are introduced into C2 that were not present in C1, apart from the CAP marker segment, and, optionally, a PRF marker segment.
- Codestream ordering. For each tile-part header of C1, there is a corresponding tile-part header in C2, and vice-versa, and all tile-part headers and packets for C2 appear in the same order as the corresponding tile-part headers and packets in C1.

NOTE 2 – There is no one-to-one mapping between the set CPF_N and the set of Rec. ITU-T T.800 | ISO/IEC 15444-1 codestreams with profile number *N* since the termination of codeword segments is not uniquely defined in either this Specification, or in Rec. ITU-T T.800 | ISO/IEC 15444-1.

9 Media types

Annex E specifies media types as defined in IETF RFC 6838.

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

Annex A

HTJ2K codestream syntax

(This annex forms an integral part of this Recommendation | International Standard.)

A.1 General

This annex specifies the extensions and constraints to the codestream syntax specified in Rec. ITU-T T.800 | ISO/IEC 15444-1 necessary to support HT code-blocks.

Table A.1 lists the marker segments affected by this Specification.

Table A.1 – Marker segments affected by this Specification (informative)

Extensions to marker segments specified in Rec. ITU-T T.800 | ISO/IEC 15444-1	CAP, COD, COC
Constraints to marker segments specified in Rec. ITU-T T.800 | ISO/IEC 15444-1	SIZ, RGN
Marker segments specified in this Recommendation | International Standard	CPF

Unless specified otherwise in clauses A.2 to A.6, an HTJ2K codestream syntax shall conform to Annex A of Rec. ITU-T T.800 | ISO/IEC 15444-1, together with any other signalled capability.

In the tables of this annex, the symbol "r" denotes bits that are reserved, and the symbol "x" denotes bits whose value can be either 0 or 1.

For HTJ2K codestreams conforming to this Specification, the value of each bit denoted with an "r" shall be 0.

NOTE – The behaviour of implementations that conform to this Specification is left unspecified when processing an HTJ2K codestream where the value of any bit denoted with an "r" is not 0.

A.2 SIZ marker segment

Bit 14 of Rsiz shall be equal to 1.

A.3 CAP marker segment

A.3.1 General

The CAP marker segment shall be present.

The value of $Pcap^{15}$ shall be equal to 1.

NOTE 1 – $Pcap^{15}$ is the 15^{th} most significant bit of the Pcap field.

Table A.2 defines values for the $Ccap^{15}$ field.

NOTE 2 – The $Ccap^{15}$ field contains information that allows a decoder to fast-fail gracefully, optimize its throughput, or generally simplify its operations, without requiring the codestream to be processed in its entirety.

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

Table A.2 – Ccap[15] syntax and semantics

Values (bits) MSB LSB	Capability
00xx xxxx xxxx xxxx	All code-blocks are HT code-blocks.
10xx xxxx xxxx xxxx	Each tile-component either consists entirely of HT code-blocks, or consists entirely of code-blocks conforming to Rec. ITU-T T.800 | ISO/IEC 15444-1.
11xx xxxx xxxx xxxx	Code-blocks within a tile-component can either be HT code-blocks, or conform to Rec. ITU-T T.800 | ISO/IEC 15444-1.
01xx xxxx xxxx xxxx	Reserved for future use by ITU-T | ISO/IEC
xx0x xxxx xxxx xxxx	Zero or one HT set is present for any HT code-block.
xx1x xxxx xxxx xxxx	More than one HT sets can be present for an HT code-block, indicating that the codestream, when decoded, can result in different quality reconstructions (see Annex B).
xxx0 xxxx xxxx xxxx	No region-of-interest marker present
xxx1 xxxx xxxx xxxx	Region-of-interest marker can be present
xxxx 0xxx xxxx xxxx	Homogeneous codestream
xxxx 1xxx xxxx xxxx	Heterogeneous codestream
xxxx xxxx xx0x xxxx	HT code-blocks only used with reversible transforms
xxxx xxxx xx1x xxxx	HT code-blocks can be used with irreversible transforms
xxxx xxxx xxxp pppp	Bits p^i specify the HT cleanup magnitude bound
xxxx xrrr rrxx xxxx	Reserved for future use by ITU-T | ISO/IEC

A.3.2 Bits 14-15 of Ccap[15]

If Bits 14 and 15 of Ccap[15] are 0, then:

- the codestream shall belong to the HTONLY set specified in clause 8.2; and
- bits 6 and 7 of all SPcod or SPcoc values are equal to 0.

If bit 14 of Ccap[15] is 0 and bit 15 of Ccap[15] is 1, then:

- the codestream shall belong to the HTDECLARED set specified in clause 8.2; and
- bit 7 of all SPcod or SPcoc values is equal to 0.

If bit 14 of Ccap[15] is 1 and bit 15 of Ccap[15] is 1, then the codestream may belong to the MIXED set specified in clause 8.2.

NOTE – A codestream that belongs to the HTONLY set can still contain SPcod or SPcoc values where both bits 6 and 7 are equal to 1.

A.3.3 Bit 13 of Ccap[15]

If bit 13 of Ccap[15] is 0, then the codestream shall belong to the SINGLEHT set specified in clause 8.3.

If bit 13 of Ccap[15] is 1, then the codestream may belong to the MULTIHT set specified in clause 8.3.

A.3.4 Bit 12 of Ccap[15]

If bit 12 of Ccap[15] is 0, then the codestream shall belong to the RGNFREE set specified in clause 8.4.

If bit 12 of Ccap[15] is 1, then the codestream may belong to the RGN set specified in clause 8.4.

A.3.5 Bit 11 of Ccap[15]

If bit 11 of Ccap[15] is 0, then the codestream shall belong to the HOMOGENEOUS set specified in clause 8.5.

If bit 11 of Ccap[15] is 1, then the codestream may belong to the HETEROGENEOUS set specified in clause 8.5.

A.3.6 Bit 5 of Ccap[15]

If bit 5 of Ccap[15] is 0, then:

- the codestream shall belong to the HTREV set specified in clause 8.6; and
- tile-components for which an irreversible transform is signalled shall have bit 6 of the SPcod or SPcoc value equal to 0.

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

If bit 5 of $Ccap^{15}$ is 1, then the codestream may belong to the HTIRV set specified in clause 8.6.

A.3.7 Bits 0-4 of $Ccap^{15}$

The codestream shall belong to the $MAGB_P$ set specified in clause 8.7.3, with the parameter B equal to:

$$B = \begin{cases} 8, & P = 0 \\ P + 8, & P < 20 \\ 4(P - 19) + 27, & 20 \leq P < 31 \\ 74, & P = 31 \end{cases}$$

where

$$P = \sum_{i=0}^{4} \mathrm{Ccap}_i^{15} \cdot 2^i \text{ , and}$$

Ccap_i^{15} is bit i of Ccap^{15} with $i = 0$ corresponding to the LSB.

NOTE – Upper bounds on the HT cleanup magnitudes μ_n can also be determined from quantization parameters found in QCD and QCC marker segments, possibly modified by the presence of RGN marker segments. These bounds can be smaller than the bound signalled by bits 0-4 of $Ccap^{15}$.

A.4 COD and COC marker segments

The COD and COC marker segments defined in Rec. ITU-T T.800 | ISO/IEC 15444-1 are modified as follows.

For a given SPcod or SPcoc value, if bit 6 is equal to 0:

- no code-blocks within the corresponding tile-component shall be HT code-blocks; and
- the semantics of the SPcod or SPcoc value are as defined in Table A.19 at Rec. ITU-T T.800 | ISO/IEC 15444-1.

Table A.3 – SPcod and SPcoc parameters semantics when bits 6 and 7 are 1 and 0, respectively

Value (bits) MSB LSB	Code-block style
01rr 0rrr	No vertically causal context
01rr 1rrr	Vertically causal context

For a given SPcod or SPcoc value, if bit 6 is equal to 1 and bit 7 is equal to 0:

- all code-blocks within the corresponding tile-component shall be HT code-blocks as defined in Annex B; and
- the semantics of the SPcod or SPcoc value shall be as defined in Table A.3.

Table A.4 – SPcod and SPcoc parameters semantics when bits 6 and 7 are 1

Value (bits) MSB LSB	Code-block style
11xx xr0r	No reset of context probabilities on coding pass boundaries (does not apply to HT code-blocks)
11xx xr1r	Reset context probabilities on coding pass boundaries (does not apply to HT code-blocks)
11xx 0rxr	No vertically causal context (applies to both Rec. ITU-T T.800 | ISO/IEC 15444-1 and HT code-blocks)
11xx 1rxr	Vertically causal context (applies to both Rec. ITU-T T.800 | ISO/IEC 15444-1 and HT code-blocks)
11x0 xrxr	No predictable termination (does not apply to HT code-blocks)
11x1 xrxr	Predictable termination (does not apply to HT code-blocks)
110x xrxr	No segmentation symbols are used (does not apply to HT code-blocks)
111x xrxr	Segmentation symbols are used (does not apply to HT code-blocks)

For a given SPcod or SPcoc value, if bit 6 is equal to 1 and bit 7 is equal to 1:

- zero or more of the code-blocks within the corresponding tile-component shall be HT code-blocks as defined in Annex B, and the remaining code-blocks shall conform to Rec. ITU-T T.800 | ISO/IEC 15444-1;

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

- if a code-block is an HT code-block, and given its first non-zero length codeword segment:
  - the first bit of that codeword segment length, as defined in clause B.10.7.1 of ITU-T T.800 | ISO/IEC 15444-1, shall be 0; and
  - *Lblock*, as defined in clause B.10.7.1 of ITU-T T.800 | ISO/IEC 15444-1, shall be greater than 3.
- Rec. ITU-T T.800 | ISO/IEC 15444-1 code-blocks shall use neither selective arithmetic coding bypass nor termination on each coding pass; and
- the semantics of the SPcod or SPcoc value shall be as defined in Table A.4.

NOTE – An HT code-block can be differentiated from a Rec. ITU-T T.800 | ISO/IEC 15444-1 code-block by processing the code-block assuming it conforms to Annex B. Failure of such processing indicates that the code-block might conform to Rec. ITU-T T.800 | ISO/IEC 15444-1.

A.5 RGN marker segment

If the RGN marker segment is present, the value of the SPrgn parameter shall be less than or equal to 37.

A.6 CPF marker segment

Function: The corresponding profile (CPF) marker segment is provided to facilitate the reversible transcoding of HTJ2K codestreams to and from codestreams that conform to Rec. ITU-T T.800 | ISO/IEC 15444-1.

Zero or one CPF marker segment shall be present in an HTJ2K codestream.

If the CPF marker segment is present, the HTJ2K codestream shall be in the set CPF_X, as specified in clause 8.8, with X equal to the CPFnum parameter of the CPF marker segment.

NOTE – An HTJ2K codestream that contains a CPF marker segment is subject to the constraints specified in the Ccap[15] field of the CAP marker segment, and by any profile signalled in the PRF marker segment.

CPFnum shall be equal to the value found in bits 0 to 11 of Rsiz of the corresponding codestream, unless that value is 4095, in which case CPFnum shall be equal to the PRFnum value found in the PRF marker segment of the corresponding codestream.

CPFnum is computed from the $Pcpf^i$ integers as follows:

$$\text{CPFnum} = -1 + \sum_{i=1}^{N} Pcpf^i \cdot 2^{16\cdot(i-1)}$$

Usage: Optional. If present, the CPF marker segment shall appear after the SIZ marker segment, CAP marker segment and, if present, the PRF marker segment, but before any other marker segments defined in Rec. ITU-T T.800 | ISO/IEC 15444-1.

Length: Variable. See Figure A.1.

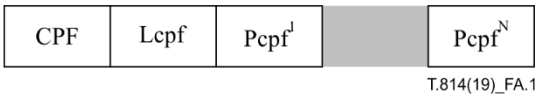


Figure A.1 – Corresponding profiles syntax

CPF Marker Code.

Lcpf Length in bytes of the CPF marker segment (not including the marker). *L*cpf is given by the following formula:

$$Lcpf = 2 + 2N$$

where N, the number of Pcpf^i values used to express *CPFnum*, is given by:

$$N = \left\lfloor \frac{\log_2(1 + \text{CPFnum})}{16} \right\rfloor + 1$$

Pcpf^i *P*cpf^i are 16-bit integers that encode CPFnum. *P*cpf^N shall not be zero.

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

Table A.5 – Corresponding profile parameter values

Parameter	Size (bits)	Value
CPF	16	0xFF59
Lcpf	16	4-65534
Pcpf[i]	16	0x0000-0xFFFF

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

Annex B

HT data organization

(This annex forms an integral part of this Recommendation | International Standard.)

B.1 HT sets

As illustrated in Figure B.1, an HT code-block consists of:

- $3 \cdot P_0$ coding passes, called placeholder passes, for which no codeword segment bytes appear in the codestream;
- followed by groups of coding passes, called HT sets, each of which consists of three coding passes, except for the last HT set, which consists of 1, 2 or 3 coding passes.

NOTE – In many cases, the first coding pass contribution from a code-block to any packet will include an HT cleanup pass and have non-zero length. However, placeholder passes can be used to preserve quality layer boundaries from a codestream that was encoded using the block-coding algorithm from Rec. ITU-T T.800 | ISO/IEC 15444-1 and was subsequently transcoded. Similarly, placeholder passes can be used to provide suggested quality layer boundaries to use when transcoding the HT block-coding algorithm representation to one that uses the block-coding algorithm from Rec. ITU-T T.800 | ISO/IEC 15444-1.

The coding passes within an HT set are defined as follows:

- The first coding pass is an HT cleanup coding pass;
- If present, the second coding pass is an HT SigProp coding pass;
- If present, the third coding pass is an MagRef coding pass.

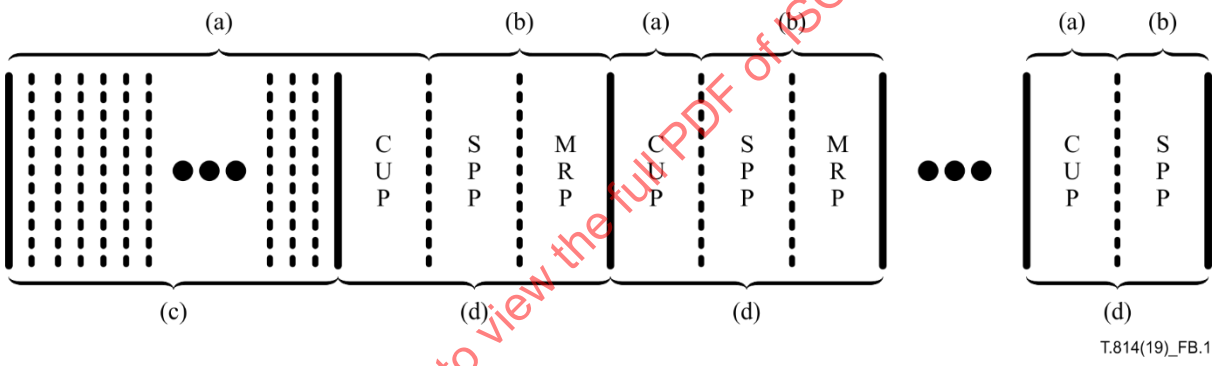


Figure B.1 – HT code-block structure. The solid vertical lines indicate HT set boundaries. The dotted lines indicate coding pass boundaries. (a) are HT cleanup segments, (b) are HT refinement segments, (c) are placeholder passes and (d) are HT sets.

B.2 HT segments

As illustrated in Figure B.1, coding passes are arranged in HT segments based on a set T of coding pass indices.

Each index in the set T defines one HT segment and corresponds to the last coding pass of the HT segment.

$$\mathrm{T} = \bigcup_{k \in \mathbb{N}} (3P_0 + T_k), \quad \text{where } T_k = \left\lceil \frac{3k}{2} \right\rceil$$

An HT cleanup segment is an HT segment that contains an HT cleanup coding pass.

An HT refinement segment is an HT segment that contains an HT SigProp coding pass.

B.3 Packets, Z_blk and S_blk

The packet, as defined in Rec. ITU-T T.800 | ISO/IEC 15444-1, that contains the first HT cleanup coding pass for a code-block shall include only one HT cleanup coding pass.

NOTE 1 – This packet can contain at most two additional coding passes: HT SigProp coding pass and HT MagRef coding pass.

Each codeword segment for a given code-block in a packet terminates with either:

- a coding pass of the code-block with an index in set T defined in clause B.2; or

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

- the last coding pass of the code-block included in the packet.

The bytes of an HT segment are obtained by concatenating the bytes of its constituent codeword segments, and the length of an HT segment is the sum of the lengths of those codeword segments.

Except for the first HT cleanup segment of a code-block, the length of an HT cleanup segment shall be either 0 or greater than 1.

The length of the first HT cleanup segment of a code-block shall be greater than 1.

If the length of an HT cleanup segment within an HT set is 0, then the length of any HT refinement segment in the same HT set shall be 0.

NOTE 2 – HT sets that contain only zero-length HT segments can be used to skip bit-planes between non-empty HT sets.

Given an HT set, Z_blk is defined as follows:

- $Z_blk = 0$, if the length of the HT cleanup segment is 0;
- $Z_blk = 1$, if the HT cleanup segment is the only segment of the HT set whose length is not 0;
- Z_blk is the number of coding passes in the HT set, otherwise.

NOTE 3 – As detailed in clause 7.1.1, Z_blk is the number of coding passes processed by the decoder. The condition where Z_blk is equal to 1 allows multiple HT cleanup coding passes to be included for a code-block, without including any SigProp or MagRef code bytes, while avoiding any concern that the empty coding passes might be decoded, impacting reconstructed image quality.

Given an HT set, the number of skipped magnitude bit-planes S_blk is defined as follows:

$$S_blk = P + P_0 + S_skip$$

where P is the number of zero-bit-planes recovered from the packet that contains the first contribution for the code-block, and S_skip is the number of HT sets preceding the given HT set.

NOTE 4 – The first contribution for the code-block can consist only of placeholder passes.

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

Annex C

CxtVLC tables

(This annex forms an integral part of this Recommendation | International Standard.)

Annex C specifies the `CxtVLC_table_0` and `CxtVLC_table_1` coding tables that are used by the `decodeCxtVLC` procedure.

The values of a coding table are specified using the bracket notation { {...}, {...} }, where inner pair of brackets are separated by commas and each inner pair of brackets specifies the values of the fields of an entry. The values of the fields are separated by commas and appear in the following order: $c_q, \rho_q, u_q^{\text{off}}, \bar{\epsilon}_q^k, \bar{\epsilon}_q^1, w, l_w$.

Hexadecimal notation is indicated by prefixing the hexadecimal number by "0x". For example, 0x41 represents an eight-bit string having only its second and its last bits (counted from the most to the LSB) equal to 1.

Numerical values not enclosed in single quotes and not prefixed by "0x" are decimal values.

The field w represents the codeword as a little-endian integer, meaning that the LSB of the integer w appears first in the VLC bit-stream, followed by the second LSB and so forth, for a total number of bits indicated by field l_w.

`CxtVLC_table_0` is specified as follows:

```
CxtVLC_table_0 = {{0, 0x1, 0x0, 0x0, 0x0, 0x06, 4},
{0, 0x1, 0x1, 0x1, 0x1, 0x3F, 7},
{0, 0x2, 0x0, 0x0, 0x0, 0x00, 3},
{0, 0x2, 0x1, 0x2, 0x2, 0x7F, 7},
{0, 0x3, 0x0, 0x0, 0x0, 0x11, 5},
{0, 0x3, 0x1, 0x2, 0x2, 0x5F, 7},
{0, 0x3, 0x1, 0x3, 0x1, 0x1F, 7},
{0, 0x4, 0x0, 0x0, 0x0, 0x02, 3},
{0, 0x4, 0x1, 0x4, 0x4, 0x13, 6},
{0, 0x5, 0x0, 0x0, 0x0, 0x0E, 5},
{0, 0x5, 0x1, 0x4, 0x4, 0x23, 6},
{0, 0x5, 0x1, 0x5, 0x1, 0x0F, 7},
{0, 0x6, 0x0, 0x0, 0x0, 0x03, 6},
{0, 0x6, 0x1, 0x0, 0x0, 0x6F, 7},
{0, 0x7, 0x0, 0x0, 0x0, 0x2F, 7},
{0, 0x7, 0x1, 0x2, 0x2, 0x4F, 7},
{0, 0x7, 0x1, 0x2, 0x0, 0x0D, 6},
{0, 0x8, 0x0, 0x0, 0x0, 0x04, 3},
{0, 0x8, 0x1, 0x8, 0x8, 0x3D, 6},
{0, 0x9, 0x0, 0x0, 0x0, 0x1D, 6},
{0, 0x9, 0x1, 0x0, 0x0, 0x2D, 6},
{0, 0xA, 0x0, 0x0, 0x0, 0x01, 5},
{0, 0xA, 0x1, 0x8, 0x8, 0x35, 6},
{0, 0xA, 0x1, 0xA, 0x2, 0x77, 7},
{0, 0xB, 0x0, 0x0, 0x0, 0x37, 7},
{0, 0xB, 0x1, 0x1, 0x1, 0x57, 7},
{0, 0xB, 0x1, 0x1, 0x0, 0x09, 6},
{0, 0xC, 0x0, 0x0, 0x0, 0x1E, 5},
```

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

```
{0, 0xC, 0x1, 0xC, 0xC, 0x17, 7},
{0, 0xC, 0x1, 0xC, 0x4, 0x15, 6},
{0, 0xC, 0x1, 0xC, 0x8, 0x25, 6},
{0, 0xD, 0x0, 0x0, 0x0, 0x67, 7},
{0, 0xD, 0x1, 0x1, 0x1, 0x27, 7},
{0, 0xD, 0x1, 0x5, 0x4, 0x47, 7},
{0, 0xD, 0x1, 0xD, 0x8, 0x07, 7},
{0, 0xE, 0x0, 0x0, 0x0, 0x7B, 7},
{0, 0xE, 0x1, 0x2, 0x2, 0x4B, 7},
{0, 0xE, 0x1, 0xA, 0x8, 0x05, 6},
{0, 0xE, 0x1, 0xE, 0x4, 0x3B, 7},
{0, 0xF, 0x0, 0x0, 0x0, 0x5B, 7},
{0, 0xF, 0x1, 0x9, 0x9, 0x1B, 7},
{0, 0xF, 0x1, 0xB, 0xA, 0x6B, 7},
{0, 0xF, 0x1, 0xF, 0xC, 0x2B, 7},
{0, 0xF, 0x1, 0xF, 0x8, 0x39, 6},
{0, 0xF, 0x1, 0xE, 0x6, 0x73, 7},
{0, 0xF, 0x1, 0xE, 0x2, 0x19, 6},
{0, 0xF, 0x1, 0xF, 0x5, 0x0B, 7},
{0, 0xF, 0x1, 0xF, 0x4, 0x29, 6},
{0, 0xF, 0x1, 0xF, 0x1, 0x33, 7},
{1, 0x0, 0x0, 0x0, 0x0, 0x00, 2},
{1, 0x1, 0x0, 0x0, 0x0, 0x0E, 4},
{1, 0x1, 0x1, 0x1, 0x1, 0x1F, 7},
{1, 0x2, 0x0, 0x0, 0x0, 0x06, 4},
{1, 0x2, 0x1, 0x2, 0x2, 0x3B, 6},
{1, 0x3, 0x0, 0x0, 0x0, 0x1B, 6},
{1, 0x3, 0x1, 0x0, 0x0, 0x3D, 6},
{1, 0x4, 0x0, 0x0, 0x0, 0x0A, 4},
{1, 0x4, 0x1, 0x4, 0x4, 0x2B, 6},
{1, 0x5, 0x0, 0x0, 0x0, 0x0B, 6},
{1, 0x5, 0x1, 0x4, 0x4, 0x33, 6},
{1, 0x5, 0x1, 0x5, 0x1, 0x7F, 7},
{1, 0x6, 0x0, 0x0, 0x0, 0x13, 6},
{1, 0x6, 0x1, 0x0, 0x0, 0x23, 6},
{1, 0x7, 0x0, 0x0, 0x0, 0x3F, 7},
{1, 0x7, 0x1, 0x2, 0x2, 0x5F, 7},
{1, 0x7, 0x1, 0x2, 0x0, 0x03, 6},
{1, 0x8, 0x0, 0x0, 0x0, 0x02, 4},
{1, 0x8, 0x1, 0x8, 0x8, 0x1D, 6},
{1, 0x9, 0x0, 0x0, 0x0, 0x2D, 6},
{1, 0x9, 0x1, 0x0, 0x0, 0x0D, 6},
```

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

```
{1, 0xA, 0x0, 0x0, 0x0, 0x35, 6},
{1, 0xA, 0x1, 0x8, 0x8, 0x15, 6},
{1, 0xA, 0x1, 0xA, 0x2, 0x6F, 7},
{1, 0xB, 0x0, 0x0, 0x0, 0x2F, 7},
{1, 0xB, 0x1, 0x1, 0x1, 0x4F, 7},
{1, 0xB, 0x1, 0x1, 0x0, 0x11, 6},
{1, 0xC, 0x0, 0x0, 0x0, 0x01, 5},
{1, 0xC, 0x1, 0x8, 0x8, 0x25, 6},
{1, 0xC, 0x1, 0xC, 0x4, 0x05, 6},
{1, 0xD, 0x0, 0x0, 0x0, 0x0F, 7},
{1, 0xD, 0x1, 0x1, 0x1, 0x17, 7},
{1, 0xD, 0x1, 0x5, 0x4, 0x39, 6},
{1, 0xD, 0x1, 0xD, 0x8, 0x77, 7},
{1, 0xE, 0x0, 0x0, 0x0, 0x37, 7},
{1, 0xE, 0x1, 0x2, 0x2, 0x57, 7},
{1, 0xE, 0x1, 0xA, 0x8, 0x19, 6},
{1, 0xE, 0x1, 0xE, 0x4, 0x67, 7},
{1, 0xF, 0x0, 0x0, 0x0, 0x07, 7},
{1, 0xF, 0x1, 0xB, 0x8, 0x29, 6},
{1, 0xF, 0x1, 0x8, 0x8, 0x27, 7},
{1, 0xF, 0x1, 0xA, 0x2, 0x09, 6},
{1, 0xF, 0x1, 0xE, 0x4, 0x31, 6},
{1, 0xF, 0x1, 0xF, 0x1, 0x47, 7},
{2, 0x0, 0x0, 0x0, 0x0, 0x00, 2},
{2, 0x1, 0x0, 0x0, 0x0, 0x0E, 4},
{2, 0x1, 0x1, 0x1, 0x1, 0x1B, 6},
{2, 0x2, 0x0, 0x0, 0x0, 0x06, 4},
{2, 0x2, 0x1, 0x2, 0x2, 0x3F, 7},
{2, 0x3, 0x0, 0x0, 0x0, 0x2B, 6},
{2, 0x3, 0x1, 0x1, 0x1, 0x33, 6},
{2, 0x3, 0x1, 0x3, 0x2, 0x7F, 7},
{2, 0x4, 0x0, 0x0, 0x0, 0x0A, 4},
{2, 0x4, 0x1, 0x4, 0x4, 0x0B, 6},
{2, 0x5, 0x0, 0x0, 0x0, 0x01, 5},
{2, 0x5, 0x1, 0x5, 0x5, 0x2F, 7},
{2, 0x5, 0x1, 0x5, 0x1, 0x13, 6},
{2, 0x5, 0x1, 0x5, 0x4, 0x23, 6},
{2, 0x6, 0x0, 0x0, 0x0, 0x03, 6},
{2, 0x6, 0x1, 0x0, 0x0, 0x5F, 7},
{2, 0x7, 0x0, 0x0, 0x0, 0x1F, 7},
{2, 0x7, 0x1, 0x2, 0x2, 0x6F, 7},
{2, 0x7, 0x1, 0x3, 0x1, 0x11, 6},
```

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

```
{2, 0x7, 0x1, 0x7, 0x4, 0x37, 7},
{2, 0x8, 0x0, 0x0, 0x0, 0x02, 4},
{2, 0x8, 0x1, 0x8, 0x8, 0x4F, 7},
{2, 0x9, 0x0, 0x0, 0x0, 0x3D, 6},
{2, 0x9, 0x1, 0x0, 0x0, 0x1D, 6},
{2, 0xA, 0x0, 0x0, 0x0, 0x2D, 6},
{2, 0xA, 0x1, 0x0, 0x0, 0x0D, 6},
{2, 0xB, 0x0, 0x0, 0x0, 0x0F, 7},
{2, 0xB, 0x1, 0x2, 0x2, 0x77, 7},
{2, 0xB, 0x1, 0x2, 0x0, 0x35, 6},
{2, 0xC, 0x0, 0x0, 0x0, 0x15, 6},
{2, 0xC, 0x1, 0x4, 0x4, 0x25, 6},
{2, 0xC, 0x1, 0xC, 0x8, 0x57, 7},
{2, 0xD, 0x0, 0x0, 0x0, 0x17, 7},
{2, 0xD, 0x1, 0x8, 0x8, 0x05, 6},
{2, 0xD, 0x1, 0xC, 0x4, 0x39, 6},
{2, 0xD, 0x1, 0xD, 0x1, 0x67, 7},
{2, 0xE, 0x0, 0x0, 0x0, 0x27, 7},
{2, 0xE, 0x1, 0x2, 0x2, 0x7B, 7},
{2, 0xE, 0x1, 0x2, 0x0, 0x19, 6},
{2, 0xF, 0x0, 0x0, 0x0, 0x47, 7},
{2, 0xF, 0x1, 0xF, 0x1, 0x29, 6},
{2, 0xF, 0x1, 0x1, 0x1, 0x09, 6},
{2, 0xF, 0x1, 0x3, 0x2, 0x07, 7},
{2, 0xF, 0x1, 0x7, 0x4, 0x31, 6},
{2, 0xF, 0x1, 0xF, 0x8, 0x3B, 7},
{3, 0x0, 0x0, 0x0, 0x0, 0x00, 3},
{3, 0x1, 0x0, 0x0, 0x0, 0x04, 4},
{3, 0x1, 0x1, 0x1, 0x1, 0x3D, 6},
{3, 0x2, 0x0, 0x0, 0x0, 0x0C, 5},
{3, 0x2, 0x1, 0x2, 0x2, 0x4F, 7},
{3, 0x3, 0x0, 0x0, 0x0, 0x1D, 6},
{3, 0x3, 0x1, 0x1, 0x1, 0x05, 6},
{3, 0x3, 0x1, 0x3, 0x2, 0x7F, 7},
{3, 0x4, 0x0, 0x0, 0x0, 0x16, 5},
{3, 0x4, 0x1, 0x4, 0x4, 0x2D, 6},
{3, 0x5, 0x0, 0x0, 0x0, 0x06, 5},
{3, 0x5, 0x1, 0x5, 0x5, 0x1A, 5},
{3, 0x5, 0x1, 0x5, 0x1, 0x0D, 6},
{3, 0x5, 0x1, 0x5, 0x4, 0x35, 6},
{3, 0x6, 0x0, 0x0, 0x0, 0x3F, 7},
{3, 0x6, 0x1, 0x4, 0x4, 0x5F, 7},
```

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

```
{3, 0x6, 0x1, 0x6, 0x2, 0x1F, 7},
{3, 0x7, 0x0, 0x0, 0x0, 0x6F, 7},
{3, 0x7, 0x1, 0x6, 0x6, 0x2F, 7},
{3, 0x7, 0x1, 0x6, 0x4, 0x15, 6},
{3, 0x7, 0x1, 0x7, 0x3, 0x77, 7},
{3, 0x7, 0x1, 0x7, 0x1, 0x25, 6},
{3, 0x7, 0x1, 0x7, 0x2, 0x0F, 7},
{3, 0x8, 0x0, 0x0, 0x0, 0x0A, 5},
{3, 0x8, 0x1, 0x8, 0x8, 0x07, 7},
{3, 0x9, 0x0, 0x0, 0x0, 0x39, 6},
{3, 0x9, 0x1, 0x1, 0x1, 0x37, 7},
{3, 0x9, 0x1, 0x9, 0x8, 0x57, 7},
{3, 0xA, 0x0, 0x0, 0x0, 0x19, 6},
{3, 0xA, 0x1, 0x8, 0x8, 0x29, 6},
{3, 0xA, 0x1, 0xA, 0x2, 0x17, 7},
{3, 0xB, 0x0, 0x0, 0x0, 0x67, 7},
{3, 0xB, 0x1, 0xB, 0x1, 0x27, 7},
{3, 0xB, 0x1, 0x1, 0x1, 0x47, 7},
{3, 0xB, 0x1, 0x3, 0x2, 0x09, 6},
{3, 0xB, 0x1, 0xB, 0x8, 0x7B, 7},
{3, 0xC, 0x0, 0x0, 0x0, 0x31, 6},
{3, 0xC, 0x1, 0x4, 0x4, 0x11, 6},
{3, 0xC, 0x1, 0xC, 0x8, 0x3B, 7},
{3, 0xD, 0x0, 0x0, 0x0, 0x5B, 7},
{3, 0xD, 0x1, 0x9, 0x9, 0x1B, 7},
{3, 0xD, 0x1, 0xD, 0x5, 0x2B, 7},
{3, 0xD, 0x1, 0xD, 0x1, 0x21, 6},
{3, 0xD, 0x1, 0xD, 0xC, 0x6B, 7},
{3, 0xD, 0x1, 0xD, 0x4, 0x01, 6},
{3, 0xD, 0x1, 0xD, 0x8, 0x4B, 7},
{3, 0xE, 0x0, 0x0, 0x0, 0x0B, 7},
{3, 0xE, 0x1, 0xE, 0x4, 0x73, 7},
{3, 0xE, 0x1, 0x4, 0x4, 0x13, 7},
{3, 0xE, 0x1, 0xC, 0x8, 0x3E, 6},
{3, 0xE, 0x1, 0xE, 0x2, 0x33, 7},
{3, 0xF, 0x0, 0x0, 0x0, 0x53, 7},
{3, 0xF, 0x1, 0xA, 0xA, 0x0E, 6},
{3, 0xF, 0x1, 0xB, 0x9, 0x63, 7},
{3, 0xF, 0x1, 0xF, 0xC, 0x03, 7},
{3, 0xF, 0x1, 0xF, 0x8, 0x12, 5},
{3, 0xF, 0x1, 0xE, 0x6, 0x23, 7},
{3, 0xF, 0x1, 0xF, 0x5, 0x1E, 6},
```

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

```
{3, 0xF, 0x1, 0xF, 0x4, 0x02, 5},
{3, 0xF, 0x1, 0xF, 0x3, 0x43, 7},
{3, 0xF, 0x1, 0xF, 0x1, 0x1C, 5},
{3, 0xF, 0x1, 0xF, 0x2, 0x2E, 6},
{4, 0x0, 0x0, 0x0, 0x0, 0x00, 2},
{4, 0x1, 0x0, 0x0, 0x0, 0x0E, 4},
{4, 0x1, 0x1, 0x1, 0x1, 0x3F, 7},
{4, 0x2, 0x0, 0x0, 0x0, 0x06, 4},
{4, 0x2, 0x1, 0x2, 0x2, 0x1B, 6},
{4, 0x3, 0x0, 0x0, 0x0, 0x2B, 6},
{4, 0x3, 0x1, 0x2, 0x2, 0x3D, 6},
{4, 0x3, 0x1, 0x3, 0x1, 0x7F, 7},
{4, 0x4, 0x0, 0x0, 0x0, 0x0A, 4},
{4, 0x4, 0x1, 0x4, 0x4, 0x5F, 7},
{4, 0x5, 0x0, 0x0, 0x0, 0x0B, 6},
{4, 0x5, 0x1, 0x0, 0x0, 0x33, 6},
{4, 0x6, 0x0, 0x0, 0x0, 0x13, 6},
{4, 0x6, 0x1, 0x0, 0x0, 0x23, 6},
{4, 0x7, 0x0, 0x0, 0x0, 0x1F, 7},
{4, 0x7, 0x1, 0x4, 0x4, 0x6F, 7},
{4, 0x7, 0x1, 0x4, 0x0, 0x03, 6},
{4, 0x8, 0x0, 0x0, 0x0, 0x02, 4},
{4, 0x8, 0x1, 0x8, 0x8, 0x1D, 6},
{4, 0x9, 0x0, 0x0, 0x0, 0x11, 6},
{4, 0x9, 0x1, 0x0, 0x0, 0x77, 7},
{4, 0xA, 0x0, 0x0, 0x0, 0x01, 5},
{4, 0xA, 0x1, 0xA, 0xA, 0x2F, 7},
{4, 0xA, 0x1, 0xA, 0x2, 0x2D, 6},
{4, 0xA, 0x1, 0xA, 0x8, 0x0D, 6},
{4, 0xB, 0x0, 0x0, 0x0, 0x4F, 7},
{4, 0xB, 0x1, 0xB, 0x2, 0x0F, 7},
{4, 0xB, 0x1, 0x0, 0x0, 0x35, 6},
{4, 0xC, 0x0, 0x0, 0x0, 0x15, 6},
{4, 0xC, 0x1, 0x8, 0x8, 0x25, 6},
{4, 0xC, 0x1, 0xC, 0x4, 0x37, 7},
{4, 0xD, 0x0, 0x0, 0x0, 0x57, 7},
{4, 0xD, 0x1, 0x1, 0x1, 0x07, 7},
{4, 0xD, 0x1, 0x1, 0x0, 0x05, 6},
{4, 0xE, 0x0, 0x0, 0x0, 0x17, 7},
{4, 0xE, 0x1, 0x4, 0x4, 0x39, 6},
{4, 0xE, 0x1, 0xC, 0x8, 0x19, 6},
{4, 0xE, 0x1, 0xE, 0x2, 0x67, 7},
```

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

```
{4, 0xF, 0x0, 0x0, 0x0, 0x27, 7},
{4, 0xF, 0x1, 0x9, 0x9, 0x47, 7},
{4, 0xF, 0x1, 0x9, 0x1, 0x29, 6},
{4, 0xF, 0x1, 0x7, 0x6, 0x7B, 7},
{4, 0xF, 0x1, 0x7, 0x2, 0x09, 6},
{4, 0xF, 0x1, 0xB, 0x8, 0x31, 6},
{4, 0xF, 0x1, 0xF, 0x4, 0x3B, 7},
{5, 0x0, 0x0, 0x0, 0x0, 0x00, 3},
{5, 0x1, 0x0, 0x0, 0x0, 0x1A, 5},
{5, 0x1, 0x1, 0x1, 0x1, 0x7F, 7},
{5, 0x2, 0x0, 0x0, 0x0, 0x0A, 5},
{5, 0x2, 0x1, 0x2, 0x2, 0x1D, 6},
{5, 0x3, 0x0, 0x0, 0x0, 0x2D, 6},
{5, 0x3, 0x1, 0x3, 0x3, 0x5F, 7},
{5, 0x3, 0x1, 0x3, 0x2, 0x39, 6},
{5, 0x3, 0x1, 0x3, 0x1, 0x3F, 7},
{5, 0x4, 0x0, 0x0, 0x0, 0x12, 5},
{5, 0x4, 0x1, 0x4, 0x4, 0x1F, 7},
{5, 0x5, 0x0, 0x0, 0x0, 0x0D, 6},
{5, 0x5, 0x1, 0x4, 0x4, 0x35, 6},
{5, 0x5, 0x1, 0x5, 0x1, 0x6F, 7},
{5, 0x6, 0x0, 0x0, 0x0, 0x15, 6},
{5, 0x6, 0x1, 0x2, 0x2, 0x25, 6},
{5, 0x6, 0x1, 0x6, 0x4, 0x2F, 7},
{5, 0x7, 0x0, 0x0, 0x0, 0x4F, 7},
{5, 0x7, 0x1, 0x6, 0x6, 0x57, 7},
{5, 0x7, 0x1, 0x6, 0x4, 0x05, 6},
{5, 0x7, 0x1, 0x7, 0x3, 0x0F, 7},
{5, 0x7, 0x1, 0x7, 0x2, 0x77, 7},
{5, 0x7, 0x1, 0x7, 0x1, 0x37, 7},
{5, 0x8, 0x0, 0x0, 0x0, 0x02, 5},
{5, 0x8, 0x1, 0x8, 0x8, 0x19, 6},
{5, 0x9, 0x0, 0x0, 0x0, 0x26, 6},
{5, 0x9, 0x1, 0x8, 0x8, 0x17, 7},
{5, 0x9, 0x1, 0x9, 0x1, 0x67, 7},
{5, 0xA, 0x0, 0x0, 0x0, 0x1C, 5},
{5, 0xA, 0x1, 0xA, 0xA, 0x29, 6},
{5, 0xA, 0x1, 0xA, 0x2, 0x09, 6},
{5, 0xA, 0x1, 0xA, 0x8, 0x31, 6},
{5, 0xB, 0x0, 0x0, 0x0, 0x27, 7},
{5, 0xB, 0x1, 0x9, 0x9, 0x07, 7},
{5, 0xB, 0x1, 0x9, 0x8, 0x11, 6},
```

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

```
{5, 0xB, 0x1, 0xB, 0x3, 0x47, 7},
{5, 0xB, 0x1, 0xB, 0x2, 0x21, 6},
{5, 0xB, 0x1, 0xB, 0x1, 0x7B, 7},
{5, 0xC, 0x0, 0x0, 0x0, 0x01, 6},
{5, 0xC, 0x1, 0x8, 0x8, 0x3E, 6},
{5, 0xC, 0x1, 0xC, 0x4, 0x3B, 7},
{5, 0xD, 0x0, 0x0, 0x0, 0x5B, 7},
{5, 0xD, 0x1, 0x9, 0x9, 0x6B, 7},
{5, 0xD, 0x1, 0x9, 0x8, 0x1E, 6},
{5, 0xD, 0x1, 0xD, 0x5, 0x1B, 7},
{5, 0xD, 0x1, 0xD, 0x4, 0x2E, 6},
{5, 0xD, 0x1, 0xD, 0x1, 0x2B, 7},
{5, 0xE, 0x0, 0x0, 0x0, 0x4B, 7},
{5, 0xE, 0x1, 0x6, 0x6, 0x0B, 7},
{5, 0xE, 0x1, 0xE, 0xA, 0x33, 7},
{5, 0xE, 0x1, 0xE, 0x2, 0x0E, 6},
{5, 0xE, 0x1, 0xE, 0xC, 0x73, 7},
{5, 0xE, 0x1, 0xE, 0x8, 0x36, 6},
{5, 0xE, 0x1, 0xE, 0x4, 0x53, 7},
{5, 0xF, 0x0, 0x0, 0x0, 0x13, 7},
{5, 0xF, 0x1, 0x7, 0x7, 0x43, 7},
{5, 0xF, 0x1, 0x7, 0x6, 0x16, 6},
{5, 0xF, 0x1, 0x7, 0x5, 0x63, 7},
{5, 0xF, 0x1, 0xF, 0xC, 0x23, 7},
{5, 0xF, 0x1, 0xF, 0x4, 0x0C, 5},
{5, 0xF, 0x1, 0xD, 0x9, 0x03, 7},
{5, 0xF, 0x1, 0xF, 0xA, 0x3D, 7},
{5, 0xF, 0x1, 0xF, 0x8, 0x14, 5},
{5, 0xF, 0x1, 0xF, 0x3, 0x7D, 7},
{5, 0xF, 0x1, 0xF, 0x2, 0x04, 5},
{5, 0xF, 0x1, 0xF, 0x1, 0x06, 6},
{6, 0x0, 0x0, 0x0, 0x0, 0x00, 3},
{6, 0x1, 0x0, 0x0, 0x0, 0x04, 4},
{6, 0x1, 0x1, 0x1, 0x1, 0x03, 6},
{6, 0x2, 0x0, 0x0, 0x0, 0x0C, 5},
{6, 0x2, 0x1, 0x2, 0x2, 0x0D, 6},
{6, 0x3, 0x0, 0x0, 0x0, 0x1A, 5},
{6, 0x3, 0x1, 0x3, 0x3, 0x3D, 6},
{6, 0x3, 0x1, 0x3, 0x1, 0x1D, 6},
{6, 0x3, 0x1, 0x3, 0x2, 0x2D, 6},
{6, 0x4, 0x0, 0x0, 0x0, 0x0A, 5},
{6, 0x4, 0x1, 0x4, 0x4, 0x3F, 7},
```

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

```
{6, 0x5, 0x0, 0x0, 0x0, 0x35, 6},
{6, 0x5, 0x1, 0x1, 0x1, 0x15, 6},
{6, 0x5, 0x1, 0x5, 0x4, 0x7F, 7},
{6, 0x6, 0x0, 0x0, 0x0, 0x25, 6},
{6, 0x6, 0x1, 0x2, 0x2, 0x5F, 7},
{6, 0x6, 0x1, 0x6, 0x4, 0x1F, 7},
{6, 0x7, 0x0, 0x0, 0x0, 0x6F, 7},
{6, 0x7, 0x1, 0x6, 0x6, 0x4F, 7},
{6, 0x7, 0x1, 0x6, 0x4, 0x05, 6},
{6, 0x7, 0x1, 0x7, 0x3, 0x2F, 7},
{6, 0x7, 0x1, 0x7, 0x1, 0x36, 6},
{6, 0x7, 0x1, 0x7, 0x2, 0x77, 7},
{6, 0x8, 0x0, 0x0, 0x0, 0x12, 5},
{6, 0x8, 0x1, 0x8, 0x8, 0x0F, 7},
{6, 0x9, 0x0, 0x0, 0x0, 0x39, 6},
{6, 0x9, 0x1, 0x1, 0x1, 0x37, 7},
{6, 0x9, 0x1, 0x9, 0x8, 0x57, 7},
{6, 0xA, 0x0, 0x0, 0x0, 0x19, 6},
{6, 0xA, 0x1, 0x2, 0x2, 0x29, 6},
{6, 0xA, 0x1, 0xA, 0x8, 0x17, 7},
{6, 0xB, 0x0, 0x0, 0x0, 0x67, 7},
{6, 0xB, 0x1, 0x9, 0x9, 0x47, 7},
{6, 0xB, 0x1, 0x9, 0x1, 0x09, 6},
{6, 0xB, 0x1, 0xB, 0xA, 0x27, 7},
{6, 0xB, 0x1, 0xB, 0x2, 0x31, 6},
{6, 0xB, 0x1, 0xB, 0x8, 0x7B, 7},
{6, 0xC, 0x0, 0x0, 0x0, 0x11, 6},
{6, 0xC, 0x1, 0xC, 0xC, 0x07, 7},
{6, 0xC, 0x1, 0xC, 0x8, 0x21, 6},
{6, 0xC, 0x1, 0xC, 0x4, 0x3B, 7},
{6, 0xD, 0x0, 0x0, 0x0, 0x5B, 7},
{6, 0xD, 0x1, 0x5, 0x5, 0x33, 7},
{6, 0xD, 0x1, 0x5, 0x4, 0x01, 6},
{6, 0xD, 0x1, 0xC, 0x8, 0x1B, 7},
{6, 0xD, 0x1, 0xD, 0x1, 0x6B, 7},
{6, 0xE, 0x0, 0x0, 0x0, 0x2B, 7},
{6, 0xE, 0x1, 0xE, 0x2, 0x4B, 7},
{6, 0xE, 0x1, 0x2, 0x2, 0x0B, 7},
{6, 0xE, 0x1, 0xE, 0xC, 0x73, 7},
{6, 0xE, 0x1, 0xE, 0x8, 0x3E, 6},
{6, 0xE, 0x1, 0xE, 0x4, 0x53, 7},
{6, 0xF, 0x0, 0x0, 0x0, 0x13, 7},
```

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

```
{6, 0xF, 0x1, 0x6, 0x6, 0x1E, 6},
{6, 0xF, 0x1, 0xE, 0xA, 0x2E, 6},
{6, 0xF, 0x1, 0xF, 0x3, 0x0E, 6},
{6, 0xF, 0x1, 0xF, 0x2, 0x02, 5},
{6, 0xF, 0x1, 0xB, 0x9, 0x63, 7},
{6, 0xF, 0x1, 0xF, 0xC, 0x16, 6},
{6, 0xF, 0x1, 0xF, 0x8, 0x06, 6},
{6, 0xF, 0x1, 0xF, 0x5, 0x23, 7},
{6, 0xF, 0x1, 0xF, 0x1, 0x1C, 5},
{6, 0xF, 0x1, 0xF, 0x4, 0x26, 6},
{7, 0x0, 0x0, 0x0, 0x0, 0x12, 5},
{7, 0x1, 0x0, 0x0, 0x0, 0x05, 6},
{7, 0x1, 0x1, 0x1, 0x1, 0x7F, 7},
{7, 0x2, 0x0, 0x0, 0x0, 0x39, 6},
{7, 0x2, 0x1, 0x2, 0x2, 0x3F, 7},
{7, 0x3, 0x0, 0x0, 0x0, 0x5F, 7},
{7, 0x3, 0x1, 0x3, 0x3, 0x1F, 7},
{7, 0x3, 0x1, 0x3, 0x2, 0x6F, 7},
{7, 0x3, 0x1, 0x3, 0x1, 0x2F, 7},
{7, 0x4, 0x0, 0x0, 0x0, 0x4F, 7},
{7, 0x4, 0x1, 0x4, 0x4, 0x0F, 7},
{7, 0x5, 0x0, 0x0, 0x0, 0x57, 7},
{7, 0x5, 0x1, 0x1, 0x1, 0x19, 6},
{7, 0x5, 0x1, 0x5, 0x4, 0x77, 7},
{7, 0x6, 0x0, 0x0, 0x0, 0x37, 7},
{7, 0x6, 0x1, 0x0, 0x0, 0x29, 6},
{7, 0x7, 0x0, 0x0, 0x0, 0x17, 7},
{7, 0x7, 0x1, 0x6, 0x6, 0x67, 7},
{7, 0x7, 0x1, 0x7, 0x3, 0x27, 7},
{7, 0x7, 0x1, 0x7, 0x2, 0x47, 7},
{7, 0x7, 0x1, 0x7, 0x5, 0x1B, 7},
{7, 0x7, 0x1, 0x7, 0x1, 0x09, 6},
{7, 0x7, 0x1, 0x7, 0x4, 0x07, 7},
{7, 0x8, 0x0, 0x0, 0x0, 0x7B, 7},
{7, 0x8, 0x1, 0x8, 0x8, 0x3B, 7},
{7, 0x9, 0x0, 0x0, 0x0, 0x5B, 7},
{7, 0x9, 0x1, 0x0, 0x0, 0x31, 6},
{7, 0xA, 0x0, 0x0, 0x0, 0x53, 7},
{7, 0xA, 0x1, 0x2, 0x2, 0x11, 6},
{7, 0xA, 0x1, 0xA, 0x8, 0x6B, 7},
{7, 0xB, 0x0, 0x0, 0x0, 0x2B, 7},
{7, 0xB, 0x1, 0x9, 0x9, 0x4B, 7},
```

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

```
{7, 0xB, 0x1, 0xB, 0x3, 0x0B, 7},
{7, 0xB, 0x1, 0xB, 0x1, 0x73, 7},
{7, 0xB, 0x1, 0xB, 0xA, 0x33, 7},
{7, 0xB, 0x1, 0xB, 0x2, 0x21, 6},
{7, 0xB, 0x1, 0xB, 0x8, 0x13, 7},
{7, 0xC, 0x0, 0x0, 0x0, 0x63, 7},
{7, 0xC, 0x1, 0x8, 0x8, 0x23, 7},
{7, 0xC, 0x1, 0xC, 0x4, 0x43, 7},
{7, 0xD, 0x0, 0x0, 0x0, 0x03, 7},
{7, 0xD, 0x1, 0x9, 0x9, 0x7D, 7},
{7, 0xD, 0x1, 0xD, 0x5, 0x5D, 7},
{7, 0xD, 0x1, 0xD, 0x1, 0x01, 6},
{7, 0xD, 0x1, 0xD, 0xC, 0x3D, 7},
{7, 0xD, 0x1, 0xD, 0x4, 0x3E, 6},
{7, 0xD, 0x1, 0xD, 0x8, 0x1D, 7},
{7, 0xE, 0x0, 0x0, 0x0, 0x6D, 7},
{7, 0xE, 0x1, 0x6, 0x6, 0x2D, 7},
{7, 0xE, 0x1, 0xE, 0xA, 0x0D, 7},
{7, 0xE, 0x1, 0xE, 0x2, 0x1E, 6},
{7, 0xE, 0x1, 0xE, 0xC, 0x4D, 7},
{7, 0xE, 0x1, 0xE, 0x8, 0x0E, 6},
{7, 0xE, 0x1, 0xE, 0x4, 0x75, 7},
{7, 0xF, 0x0, 0x0, 0x0, 0x15, 7},
{7, 0xF, 0x1, 0xF, 0xF, 0x06, 5},
{7, 0xF, 0x1, 0xF, 0xD, 0x35, 7},
{7, 0xF, 0x1, 0xF, 0x7, 0x55, 7},
{7, 0xF, 0x1, 0xF, 0x5, 0x1A, 5},
{7, 0xF, 0x1, 0xF, 0xB, 0x25, 7},
{7, 0xF, 0x1, 0xF, 0x3, 0x0A, 5},
{7, 0xF, 0x1, 0xF, 0x9, 0x2E, 6},
{7, 0xF, 0x1, 0xF, 0x1, 0x00, 4},
{7, 0xF, 0x1, 0xF, 0xE, 0x65, 7},
{7, 0xF, 0x1, 0xF, 0x6, 0x36, 6},
{7, 0xF, 0x1, 0xF, 0xA, 0x02, 5},
{7, 0xF, 0x1, 0xF, 0x2, 0x0C, 4},
{7, 0xF, 0x1, 0xF, 0xC, 0x16, 6},
{7, 0xF, 0x1, 0xF, 0x8, 0x04, 4},
{7, 0xF, 0x1, 0xF, 0x4, 0x08, 4}}
```

`CxtVLC_table_1` is specified as follows:

```
CxtVLC_table_1 = {{0, 0x1, 0x0, 0x0, 0x0, 0x00, 3},
{0, 0x1, 0x1, 0x1, 0x1, 0x27, 6},
```

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

```
{0, 0x2, 0x0, 0x0, 0x0, 0x06, 3},
{0, 0x2, 0x1, 0x2, 0x2, 0x17, 6},
{0, 0x3, 0x0, 0x0, 0x0, 0x0D, 5},
{0, 0x3, 0x1, 0x0, 0x0, 0x3B, 6},
{0, 0x4, 0x0, 0x0, 0x0, 0x02, 3},
{0, 0x4, 0x1, 0x4, 0x4, 0x07, 6},
{0, 0x5, 0x0, 0x0, 0x0, 0x15, 5},
{0, 0x5, 0x1, 0x0, 0x0, 0x2B, 6},
{0, 0x6, 0x0, 0x0, 0x0, 0x01, 5},
{0, 0x6, 0x1, 0x0, 0x0, 0x7F, 7},
{0, 0x7, 0x0, 0x0, 0x0, 0x1F, 7},
{0, 0x7, 0x1, 0x0, 0x0, 0x1B, 6},
{0, 0x8, 0x0, 0x0, 0x0, 0x04, 3},
{0, 0x8, 0x1, 0x8, 0x8, 0x05, 5},
{0, 0x9, 0x0, 0x0, 0x0, 0x19, 5},
{0, 0x9, 0x1, 0x0, 0x0, 0x13, 6},
{0, 0xA, 0x0, 0x0, 0x0, 0x09, 5},
{0, 0xA, 0x1, 0x8, 0x8, 0x0B, 6},
{0, 0xA, 0x1, 0xA, 0x2, 0x3F, 7},
{0, 0xB, 0x0, 0x0, 0x0, 0x5F, 7},
{0, 0xB, 0x1, 0x0, 0x0, 0x33, 6},
{0, 0xC, 0x0, 0x0, 0x0, 0x11, 5},
{0, 0xC, 0x1, 0x8, 0x8, 0x23, 6},
{0, 0xC, 0x1, 0xC, 0x4, 0x6F, 7},
{0, 0xD, 0x0, 0x0, 0x0, 0x0F, 7},
{0, 0xD, 0x1, 0x0, 0x0, 0x03, 6},
{0, 0xE, 0x0, 0x0, 0x0, 0x2F, 7},
{0, 0xE, 0x1, 0x4, 0x4, 0x4F, 7},
{0, 0xE, 0x1, 0x4, 0x0, 0x3D, 6},
{0, 0xF, 0x0, 0x0, 0x0, 0x77, 7},
{0, 0xF, 0x1, 0x1, 0x1, 0x37, 7},
{0, 0xF, 0x1, 0x1, 0x0, 0x1D, 6},
{1, 0x0, 0x0, 0x0, 0x0, 0x00, 1},
{1, 0x1, 0x0, 0x0, 0x0, 0x05, 4},
{1, 0x1, 0x1, 0x1, 0x1, 0x7F, 7},
{1, 0x2, 0x0, 0x0, 0x0, 0x09, 4},
{1, 0x2, 0x1, 0x2, 0x2, 0x1F, 7},
{1, 0x3, 0x0, 0x0, 0x0, 0x1D, 5},
{1, 0x3, 0x1, 0x1, 0x1, 0x3F, 7},
{1, 0x3, 0x1, 0x3, 0x2, 0x5F, 7},
{1, 0x4, 0x0, 0x0, 0x0, 0x0D, 5},
{1, 0x4, 0x1, 0x4, 0x4, 0x37, 7},
```

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

```
{1, 0x5, 0x0, 0x0, 0x0, 0x03, 6},
{1, 0x5, 0x1, 0x0, 0x0, 0x6F, 7},
{1, 0x6, 0x0, 0x0, 0x0, 0x2F, 7},
{1, 0x6, 0x1, 0x0, 0x0, 0x4F, 7},
{1, 0x7, 0x0, 0x0, 0x0, 0x0F, 7},
{1, 0x7, 0x1, 0x0, 0x0, 0x77, 7},
{1, 0x8, 0x0, 0x0, 0x0, 0x01, 4},
{1, 0x8, 0x1, 0x8, 0x8, 0x17, 7},
{1, 0x9, 0x0, 0x0, 0x0, 0x0B, 6},
{1, 0x9, 0x1, 0x0, 0x0, 0x57, 7},
{1, 0xA, 0x0, 0x0, 0x0, 0x33, 6},
{1, 0xA, 0x1, 0x0, 0x0, 0x67, 7},
{1, 0xB, 0x0, 0x0, 0x0, 0x27, 7},
{1, 0xB, 0x1, 0x0, 0x0, 0x2B, 7},
{1, 0xC, 0x0, 0x0, 0x0, 0x13, 6},
{1, 0xC, 0x1, 0x0, 0x0, 0x47, 7},
{1, 0xD, 0x0, 0x0, 0x0, 0x07, 7},
{1, 0xD, 0x1, 0x0, 0x0, 0x7B, 7},
{1, 0xE, 0x0, 0x0, 0x0, 0x3B, 7},
{1, 0xE, 0x1, 0x0, 0x0, 0x5B, 7},
{1, 0xF, 0x0, 0x0, 0x0, 0x1B, 7},
{1, 0xF, 0x1, 0x4, 0x4, 0x6B, 7},
{1, 0xF, 0x1, 0x4, 0x0, 0x23, 6},
{2, 0x0, 0x0, 0x0, 0x0, 0x00, 1},
{2, 0x1, 0x0, 0x0, 0x0, 0x09, 4},
{2, 0x1, 0x1, 0x1, 0x1, 0x7F, 7},
{2, 0x2, 0x0, 0x0, 0x0, 0x01, 4},
{2, 0x2, 0x1, 0x2, 0x2, 0x23, 6},
{2, 0x3, 0x0, 0x0, 0x0, 0x3D, 6},
{2, 0x3, 0x1, 0x2, 0x2, 0x3F, 7},
{2, 0x3, 0x1, 0x3, 0x1, 0x1F, 7},
{2, 0x4, 0x0, 0x0, 0x0, 0x15, 5},
{2, 0x4, 0x1, 0x4, 0x4, 0x5F, 7},
{2, 0x5, 0x0, 0x0, 0x0, 0x03, 6},
{2, 0x5, 0x1, 0x0, 0x0, 0x6F, 7},
{2, 0x6, 0x0, 0x0, 0x0, 0x2F, 7},
{2, 0x6, 0x1, 0x0, 0x0, 0x4F, 7},
{2, 0x7, 0x0, 0x0, 0x0, 0x0F, 7},
{2, 0x7, 0x1, 0x0, 0x0, 0x17, 7},
{2, 0x8, 0x0, 0x0, 0x0, 0x05, 5},
{2, 0x8, 0x1, 0x8, 0x8, 0x77, 7},
{2, 0x9, 0x0, 0x0, 0x0, 0x37, 7},
```

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

```
{2, 0x9, 0x1, 0x0, 0x0, 0x57, 7},
{2, 0xA, 0x0, 0x0, 0x0, 0x1D, 6},
{2, 0xA, 0x1, 0xA, 0xA, 0x7B, 7},
{2, 0xA, 0x1, 0xA, 0x2, 0x2D, 6},
{2, 0xA, 0x1, 0xA, 0x8, 0x67, 7},
{2, 0xB, 0x0, 0x0, 0x0, 0x27, 7},
{2, 0xB, 0x1, 0xB, 0x2, 0x47, 7},
{2, 0xB, 0x1, 0x0, 0x0, 0x07, 7},
{2, 0xC, 0x0, 0x0, 0x0, 0x0D, 6},
{2, 0xC, 0x1, 0x0, 0x0, 0x3B, 7},
{2, 0xD, 0x0, 0x0, 0x0, 0x5B, 7},
{2, 0xD, 0x1, 0x0, 0x0, 0x1B, 7},
{2, 0xE, 0x0, 0x0, 0x0, 0x6B, 7},
{2, 0xE, 0x1, 0x4, 0x4, 0x2B, 7},
{2, 0xE, 0x1, 0x4, 0x0, 0x4B, 7},
{2, 0xF, 0x0, 0x0, 0x0, 0x0B, 7},
{2, 0xF, 0x1, 0x4, 0x4, 0x73, 7},
{2, 0xF, 0x1, 0x5, 0x1, 0x33, 7},
{2, 0xF, 0x1, 0x7, 0x2, 0x53, 7},
{2, 0xF, 0x1, 0xF, 0x8, 0x13, 7},
{3, 0x0, 0x0, 0x0, 0x0, 0x00, 2},
{3, 0x1, 0x0, 0x0, 0x0, 0x0A, 4},
{3, 0x1, 0x1, 0x1, 0x1, 0x0B, 6},
{3, 0x2, 0x0, 0x0, 0x0, 0x02, 4},
{3, 0x2, 0x1, 0x2, 0x2, 0x23, 6},
{3, 0x3, 0x0, 0x0, 0x0, 0x0E, 5},
{3, 0x3, 0x1, 0x3, 0x3, 0x7F, 7},
{3, 0x3, 0x1, 0x3, 0x2, 0x33, 6},
{3, 0x3, 0x1, 0x3, 0x1, 0x13, 6},
{3, 0x4, 0x0, 0x0, 0x0, 0x16, 5},
{3, 0x4, 0x1, 0x4, 0x4, 0x3F, 7},
{3, 0x5, 0x0, 0x0, 0x0, 0x03, 6},
{3, 0x5, 0x1, 0x1, 0x1, 0x3D, 6},
{3, 0x5, 0x1, 0x5, 0x4, 0x1F, 7},
{3, 0x6, 0x0, 0x0, 0x0, 0x1D, 6},
{3, 0x6, 0x1, 0x0, 0x0, 0x5F, 7},
{3, 0x7, 0x0, 0x0, 0x0, 0x2D, 6},
{3, 0x7, 0x1, 0x4, 0x4, 0x2F, 7},
{3, 0x7, 0x1, 0x5, 0x1, 0x1E, 6},
{3, 0x7, 0x1, 0x7, 0x2, 0x6F, 7},
{3, 0x8, 0x0, 0x0, 0x0, 0x06, 5},
{3, 0x8, 0x1, 0x8, 0x8, 0x4F, 7},
```

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

```
{3, 0x9, 0x0, 0x0, 0x0, 0x0D, 6},
{3, 0x9, 0x1, 0x0, 0x0, 0x35, 6},
{3, 0xA, 0x0, 0x0, 0x0, 0x15, 6},
{3, 0xA, 0x1, 0x2, 0x2, 0x25, 6},
{3, 0xA, 0x1, 0xA, 0x8, 0x0F, 7},
{3, 0xB, 0x0, 0x0, 0x0, 0x05, 6},
{3, 0xB, 0x1, 0x8, 0x8, 0x39, 6},
{3, 0xB, 0x1, 0xB, 0x3, 0x17, 7},
{3, 0xB, 0x1, 0xB, 0x2, 0x19, 6},
{3, 0xB, 0x1, 0xB, 0x1, 0x77, 7},
{3, 0xC, 0x0, 0x0, 0x0, 0x29, 6},
{3, 0xC, 0x1, 0x0, 0x0, 0x09, 6},
{3, 0xD, 0x0, 0x0, 0x0, 0x37, 7},
{3, 0xD, 0x1, 0x4, 0x4, 0x57, 7},
{3, 0xD, 0x1, 0x4, 0x0, 0x31, 6},
{3, 0xE, 0x0, 0x0, 0x0, 0x67, 7},
{3, 0xE, 0x1, 0x4, 0x4, 0x27, 7},
{3, 0xE, 0x1, 0xC, 0x8, 0x47, 7},
{3, 0xE, 0x1, 0xE, 0x2, 0x6B, 7},
{3, 0xF, 0x0, 0x0, 0x0, 0x11, 6},
{3, 0xF, 0x1, 0x6, 0x6, 0x07, 7},
{3, 0xF, 0x1, 0x7, 0x3, 0x7B, 7},
{3, 0xF, 0x1, 0xF, 0xA, 0x3B, 7},
{3, 0xF, 0x1, 0xF, 0x2, 0x21, 6},
{3, 0xF, 0x1, 0xF, 0x8, 0x01, 6},
{3, 0xF, 0x1, 0xA, 0x8, 0x5B, 7},
{3, 0xF, 0x1, 0xF, 0x5, 0x1B, 7},
{3, 0xF, 0x1, 0xF, 0x1, 0x3E, 6},
{3, 0xF, 0x1, 0xF, 0x4, 0x2B, 7},
{4, 0x0, 0x0, 0x0, 0x0, 0x00, 1},
{4, 0x1, 0x0, 0x0, 0x0, 0x0D, 5},
{4, 0x1, 0x1, 0x1, 0x1, 0x7F, 7},
{4, 0x2, 0x0, 0x0, 0x0, 0x15, 5},
{4, 0x2, 0x1, 0x2, 0x2, 0x3F, 7},
{4, 0x3, 0x0, 0x0, 0x0, 0x5F, 7},
{4, 0x3, 0x1, 0x0, 0x0, 0x6F, 7},
{4, 0x4, 0x0, 0x0, 0x0, 0x09, 4},
{4, 0x4, 0x1, 0x4, 0x4, 0x23, 6},
{4, 0x5, 0x0, 0x0, 0x0, 0x33, 6},
{4, 0x5, 0x1, 0x0, 0x0, 0x1F, 7},
{4, 0x6, 0x0, 0x0, 0x0, 0x13, 6},
{4, 0x6, 0x1, 0x0, 0x0, 0x2F, 7},
```

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

```
{4, 0x7, 0x0, 0x0, 0x0, 0x4F, 7},
{4, 0x7, 0x1, 0x0, 0x0, 0x57, 7},
{4, 0x8, 0x0, 0x0, 0x0, 0x01, 4},
{4, 0x8, 0x1, 0x8, 0x8, 0x0F, 7},
{4, 0x9, 0x0, 0x0, 0x0, 0x77, 7},
{4, 0x9, 0x1, 0x0, 0x0, 0x37, 7},
{4, 0xA, 0x0, 0x0, 0x0, 0x1D, 6},
{4, 0xA, 0x1, 0x0, 0x0, 0x17, 7},
{4, 0xB, 0x0, 0x0, 0x0, 0x67, 7},
{4, 0xB, 0x1, 0x0, 0x0, 0x6B, 7},
{4, 0xC, 0x0, 0x0, 0x0, 0x05, 5},
{4, 0xC, 0x1, 0xC, 0xC, 0x27, 7},
{4, 0xC, 0x1, 0xC, 0x8, 0x47, 7},
{4, 0xC, 0x1, 0xC, 0x4, 0x07, 7},
{4, 0xD, 0x0, 0x0, 0x0, 0x7B, 7},
{4, 0xD, 0x1, 0x0, 0x0, 0x3B, 7},
{4, 0xE, 0x0, 0x0, 0x0, 0x5B, 7},
{4, 0xE, 0x1, 0x2, 0x2, 0x1B, 7},
{4, 0xE, 0x1, 0x2, 0x0, 0x03, 6},
{4, 0xF, 0x0, 0x0, 0x0, 0x2B, 7},
{4, 0xF, 0x1, 0x1, 0x1, 0x4B, 7},
{4, 0xF, 0x1, 0x3, 0x2, 0x0B, 7},
{4, 0xF, 0x1, 0x3, 0x0, 0x3D, 6},
{5, 0x0, 0x0, 0x0, 0x0, 0x00, 2},
{5, 0x1, 0x0, 0x0, 0x0, 0x1E, 5},
{5, 0x1, 0x1, 0x1, 0x1, 0x3B, 6},
{5, 0x2, 0x0, 0x0, 0x0, 0x0A, 5},
{5, 0x2, 0x1, 0x2, 0x2, 0x3F, 7},
{5, 0x3, 0x0, 0x0, 0x0, 0x1B, 6},
{5, 0x3, 0x1, 0x0, 0x0, 0x0B, 6},
{5, 0x4, 0x0, 0x0, 0x0, 0x02, 4},
{5, 0x4, 0x1, 0x4, 0x4, 0x2B, 6},
{5, 0x5, 0x0, 0x0, 0x0, 0x0E, 5},
{5, 0x5, 0x1, 0x4, 0x4, 0x33, 6},
{5, 0x5, 0x1, 0x5, 0x1, 0x7F, 7},
{5, 0x6, 0x0, 0x0, 0x0, 0x13, 6},
{5, 0x6, 0x1, 0x0, 0x0, 0x6F, 7},
{5, 0x7, 0x0, 0x0, 0x0, 0x23, 6},
{5, 0x7, 0x1, 0x2, 0x2, 0x5F, 7},
{5, 0x7, 0x1, 0x2, 0x0, 0x15, 6},
{5, 0x8, 0x0, 0x0, 0x0, 0x16, 5},
{5, 0x8, 0x1, 0x8, 0x8, 0x03, 6},
```

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

```
{5, 0x9, 0x0, 0x0, 0x0, 0x3D, 6},
{5, 0x9, 0x1, 0x0, 0x0, 0x1F, 7},
{5, 0xA, 0x0, 0x0, 0x0, 0x1D, 6},
{5, 0xA, 0x1, 0x0, 0x0, 0x2D, 6},
{5, 0xB, 0x0, 0x0, 0x0, 0x0D, 6},
{5, 0xB, 0x1, 0x1, 0x1, 0x4F, 7},
{5, 0xB, 0x1, 0x1, 0x0, 0x35, 6},
{5, 0xC, 0x0, 0x0, 0x0, 0x06, 5},
{5, 0xC, 0x1, 0x4, 0x4, 0x25, 6},
{5, 0xC, 0x1, 0xC, 0x8, 0x2F, 7},
{5, 0xD, 0x0, 0x0, 0x0, 0x05, 6},
{5, 0xD, 0x1, 0x1, 0x1, 0x77, 7},
{5, 0xD, 0x1, 0x5, 0x4, 0x39, 6},
{5, 0xD, 0x1, 0xD, 0x8, 0x0F, 7},
{5, 0xE, 0x0, 0x0, 0x0, 0x19, 6},
{5, 0xE, 0x1, 0x2, 0x2, 0x57, 7},
{5, 0xE, 0x1, 0xA, 0x8, 0x01, 6},
{5, 0xE, 0x1, 0xE, 0x4, 0x37, 7},
{5, 0xF, 0x0, 0x0, 0x0, 0x1A, 5},
{5, 0xF, 0x1, 0x9, 0x9, 0x17, 7},
{5, 0xF, 0x1, 0xD, 0x5, 0x67, 7},
{5, 0xF, 0x1, 0xF, 0x3, 0x07, 7},
{5, 0xF, 0x1, 0xF, 0x1, 0x29, 6},
{5, 0xF, 0x1, 0x7, 0x6, 0x27, 7},
{5, 0xF, 0x1, 0xF, 0xC, 0x09, 6},
{5, 0xF, 0x1, 0xF, 0x4, 0x31, 6},
{5, 0xF, 0x1, 0xF, 0xA, 0x47, 7},
{5, 0xF, 0x1, 0xF, 0x8, 0x11, 6},
{5, 0xF, 0x1, 0xF, 0x2, 0x21, 6},
{6, 0x0, 0x0, 0x0, 0x0, 0x00, 3},
{6, 0x1, 0x0, 0x0, 0x0, 0x02, 4},
{6, 0x1, 0x1, 0x1, 0x1, 0x03, 6},
{6, 0x2, 0x0, 0x0, 0x0, 0x0C, 4},
{6, 0x2, 0x1, 0x2, 0x2, 0x3D, 6},
{6, 0x3, 0x0, 0x0, 0x0, 0x1D, 6},
{6, 0x3, 0x1, 0x2, 0x2, 0x0D, 6},
{6, 0x3, 0x1, 0x3, 0x1, 0x7F, 7},
{6, 0x4, 0x0, 0x0, 0x0, 0x04, 4},
{6, 0x4, 0x1, 0x4, 0x4, 0x2D, 6},
{6, 0x5, 0x0, 0x0, 0x0, 0x0A, 5},
{6, 0x5, 0x1, 0x4, 0x4, 0x35, 6},
{6, 0x5, 0x1, 0x5, 0x1, 0x2F, 7},
```

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

```
{6, 0x6, 0x0, 0x0, 0x0, 0x15, 6},
{6, 0x6, 0x1, 0x2, 0x2, 0x3F, 7},
{6, 0x6, 0x1, 0x6, 0x4, 0x5F, 7},
{6, 0x7, 0x0, 0x0, 0x0, 0x25, 6},
{6, 0x7, 0x1, 0x2, 0x2, 0x29, 6},
{6, 0x7, 0x1, 0x3, 0x1, 0x1F, 7},
{6, 0x7, 0x1, 0x7, 0x4, 0x6F, 7},
{6, 0x8, 0x0, 0x0, 0x0, 0x16, 5},
{6, 0x8, 0x1, 0x8, 0x8, 0x05, 6},
{6, 0x9, 0x0, 0x0, 0x0, 0x39, 6},
{6, 0x9, 0x1, 0x0, 0x0, 0x19, 6},
{6, 0xA, 0x0, 0x0, 0x0, 0x06, 5},
{6, 0xA, 0x1, 0xA, 0xA, 0x0F, 7},
{6, 0xA, 0x1, 0xA, 0x2, 0x09, 6},
{6, 0xA, 0x1, 0xA, 0x8, 0x4F, 7},
{6, 0xB, 0x0, 0x0, 0x0, 0x0E, 6},
{6, 0xB, 0x1, 0xB, 0x2, 0x77, 7},
{6, 0xB, 0x1, 0x2, 0x2, 0x37, 7},
{6, 0xB, 0x1, 0xA, 0x8, 0x57, 7},
{6, 0xB, 0x1, 0xB, 0x1, 0x47, 7},
{6, 0xC, 0x0, 0x0, 0x0, 0x1A, 5},
{6, 0xC, 0x1, 0xC, 0xC, 0x17, 7},
{6, 0xC, 0x1, 0xC, 0x8, 0x67, 7},
{6, 0xC, 0x1, 0xC, 0x4, 0x27, 7},
{6, 0xD, 0x0, 0x0, 0x0, 0x31, 6},
{6, 0xD, 0x1, 0xD, 0x4, 0x07, 7},
{6, 0xD, 0x1, 0x4, 0x4, 0x7B, 7},
{6, 0xD, 0x1, 0xC, 0x8, 0x3B, 7},
{6, 0xD, 0x1, 0xD, 0x1, 0x2B, 7},
{6, 0xE, 0x0, 0x0, 0x0, 0x11, 6},
{6, 0xE, 0x1, 0xE, 0x4, 0x5B, 7},
{6, 0xE, 0x1, 0x4, 0x4, 0x1B, 7},
{6, 0xE, 0x1, 0xE, 0xA, 0x6B, 7},
{6, 0xE, 0x1, 0xE, 0x8, 0x21, 6},
{6, 0xE, 0x1, 0xE, 0x2, 0x33, 7},
{6, 0xF, 0x0, 0x0, 0x0, 0x01, 6},
{6, 0xF, 0x1, 0x3, 0x3, 0x4B, 7},
{6, 0xF, 0x1, 0x7, 0x6, 0x0B, 7},
{6, 0xF, 0x1, 0xF, 0xA, 0x73, 7},
{6, 0xF, 0x1, 0xF, 0x2, 0x3E, 6},
{6, 0xF, 0x1, 0xB, 0x9, 0x53, 7},
{6, 0xF, 0x1, 0xF, 0xC, 0x63, 7},
```

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

```
{6, 0xF, 0x1, 0xF, 0x8, 0x1E, 6},
{6, 0xF, 0x1, 0xF, 0x5, 0x13, 7},
{6, 0xF, 0x1, 0xF, 0x4, 0x2E, 6},
{6, 0xF, 0x1, 0xF, 0x1, 0x23, 7},
{7, 0x0, 0x0, 0x0, 0x0, 0x04, 4},
{7, 0x1, 0x0, 0x0, 0x0, 0x33, 6},
{7, 0x1, 0x1, 0x1, 0x1, 0x13, 6},
{7, 0x2, 0x0, 0x0, 0x0, 0x23, 6},
{7, 0x2, 0x1, 0x2, 0x2, 0x7F, 7},
{7, 0x3, 0x0, 0x0, 0x0, 0x03, 6},
{7, 0x3, 0x1, 0x1, 0x1, 0x3F, 7},
{7, 0x3, 0x1, 0x3, 0x2, 0x6F, 7},
{7, 0x4, 0x0, 0x0, 0x0, 0x2D, 6},
{7, 0x4, 0x1, 0x4, 0x4, 0x5F, 7},
{7, 0x5, 0x0, 0x0, 0x0, 0x16, 5},
{7, 0x5, 0x1, 0x1, 0x1, 0x3D, 6},
{7, 0x5, 0x1, 0x5, 0x4, 0x1F, 7},
{7, 0x6, 0x0, 0x0, 0x0, 0x1D, 6},
{7, 0x6, 0x1, 0x0, 0x0, 0x77, 7},
{7, 0x7, 0x0, 0x0, 0x0, 0x06, 5},
{7, 0x7, 0x1, 0x7, 0x4, 0x2F, 7},
{7, 0x7, 0x1, 0x4, 0x4, 0x4F, 7},
{7, 0x7, 0x1, 0x7, 0x3, 0x0F, 7},
{7, 0x7, 0x1, 0x7, 0x1, 0x0D, 6},
{7, 0x7, 0x1, 0x7, 0x2, 0x57, 7},
{7, 0x8, 0x0, 0x0, 0x0, 0x35, 6},
{7, 0x8, 0x1, 0x8, 0x8, 0x37, 7},
{7, 0x9, 0x0, 0x0, 0x0, 0x15, 6},
{7, 0x9, 0x1, 0x0, 0x0, 0x27, 7},
{7, 0xA, 0x0, 0x0, 0x0, 0x25, 6},
{7, 0xA, 0x1, 0x0, 0x0, 0x29, 6},
{7, 0xB, 0x0, 0x0, 0x0, 0x1A, 5},
{7, 0xB, 0x1, 0xB, 0x1, 0x17, 7},
{7, 0xB, 0x1, 0x1, 0x1, 0x67, 7},
{7, 0xB, 0x1, 0x3, 0x2, 0x05, 6},
{7, 0xB, 0x1, 0xB, 0x8, 0x7B, 7},
{7, 0xC, 0x0, 0x0, 0x0, 0x39, 6},
{7, 0xC, 0x1, 0x0, 0x0, 0x19, 6},
{7, 0xD, 0x0, 0x0, 0x0, 0x0C, 5},
{7, 0xD, 0x1, 0xD, 0x1, 0x47, 7},
{7, 0xD, 0x1, 0x1, 0x1, 0x07, 7},
{7, 0xD, 0x1, 0x5, 0x4, 0x09, 6},
```

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

```
{7, 0xD, 0x1, 0xD, 0x8, 0x1B, 7},
{7, 0xE, 0x0, 0x0, 0x0, 0x31, 6},
{7, 0xE, 0x1, 0xE, 0x2, 0x3B, 7},
{7, 0xE, 0x1, 0x2, 0x2, 0x5B, 7},
{7, 0xE, 0x1, 0xA, 0x8, 0x3E, 6},
{7, 0xE, 0x1, 0xE, 0x4, 0x0B, 7},
{7, 0xF, 0x0, 0x0, 0x0, 0x00, 3},
{7, 0xF, 0x1, 0xF, 0xF, 0x6B, 7},
{7, 0xF, 0x1, 0xF, 0x7, 0x2B, 7},
{7, 0xF, 0x1, 0xF, 0xB, 0x4B, 7},
{7, 0xF, 0x1, 0xF, 0x3, 0x11, 6},
{7, 0xF, 0x1, 0x7, 0x6, 0x21, 6},
{7, 0xF, 0x1, 0xF, 0xA, 0x01, 6},
{7, 0xF, 0x1, 0xF, 0x2, 0x0A, 5},
{7, 0xF, 0x1, 0xB, 0x9, 0x1E, 6},
{7, 0xF, 0x1, 0xF, 0xC, 0x0E, 6},
{7, 0xF, 0x1, 0xF, 0x8, 0x12, 5},
{7, 0xF, 0x1, 0xF, 0x5, 0x2E, 6},
{7, 0xF, 0x1, 0xF, 0x1, 0x02, 5},
{7, 0xF, 0x1, 0xF, 0x4, 0x1C, 5}}
```

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

Annex D

JPH file format

(This annex forms an integral part of this Recommendation | International Standard.)

D.1 General

The JPH file format conforms to the JP2 file format specified in Rec. ITU-T T.800 | ISO/IEC 15444-1, unless specified otherwise in this annex.

D.2 JP2 Header box

In contrast to the JP2 file format, if the `UnkC` field is non-zero it is not required that a Colour Specification box be present within the JP2 Header box.

If the JP2 Header box does not contain a Colour Specification box:

- the colourspace of the image data is unspecified; and
- no Typ^{i} field shall be equal to 0.

D.3 File Type box

The `BR` field shall be equal to `'jph\040'`.

The `MinV` field shall be 0.

One `CLi` field shall be equal to the value `'jph\040'`.

D.4 Colour Specification box

D.4.1 Additional METH values

This standard defines the METH values listed in Table D.1.

Table D.1 – Additional METH values

Value	Meaning
0-2	As specified in Rec. ITU-T T.800 | ISO/IEC 15444-1.
3	Any International Color Consortium (ICC) method. This Colour Specification box indicates that the colourspace of the codestream is specified by an embedded input ICC profile. Contrary to the Restricted ICC method defined in the JP2 file format, this method allows for any input ICC profile, described in ISO/IEC 15076-1.
5	Parameterized colourspace as specified in Rec. ITU-T H.273 | ISO/IEC 23001-8

D.4.2 Any International Color Consortium method

When the METH field is equal to 3, the Colour Specification box shall be organized as specified in Figure D.1 and Table D.2.

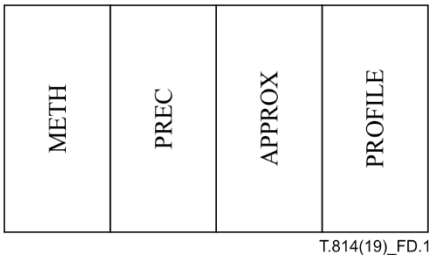


Figure D.1 – Organization of the contents of a Colour Specification box when METH = 3

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

Table D.2 – Format of the contents of the Colour Specification box

Field name	Size (bits)	Value
PROFILE	Variable	ICC input profile as defined by ISO/IEC 15076-1, specifying the transformation between the decompressed code values and the PCS. Any input ICC profile, regardless of profile class, may be contained within this field.

NOTE – This method is equivalent to the Any ICC method specified in Rec. ITU-T T.801 | ISO/IEC 15444-2.

D.4.3 Parameterized colourspace

When the METH field is equal to 5, the Colour Specification box shall be organized as specified in Table D.3 and Figure D.2.

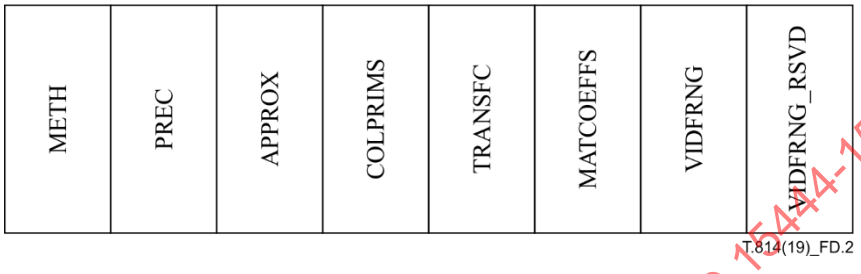


Figure D.2 – Organization of the contents of a Colour Specification box when METH = 5

Table D.3 – Format of the contents of the Colour Specification box

Field name	Size (bits)	Value
COLPRIMS	16	One of the ColourPrimaries enumerated values specified in Rec. ITU-T H.273 | ISO/IEC 23001-8
TRANSFC	16	One of the TransferCharacteristics enumerated values specified in Rec. ITU-T H.273 | ISO/IEC 23001-8
MATCOEFFS	16	One of the MatrixCoefficients enumerated values specified in Rec. ITU-T H.273 | ISO/IEC 23001-8
VIDFRNG	1	Value of the VideoFullRangeFlag specified in Rec. ITU-T H.273 | ISO/IEC 23001-8
VIDFRNG _RSVD	7	Reserved for future use by ITU-T | ISO/IEC

D.5 Contiguous codestream box

The Contiguous Codestream box shall contains a valid and complete HTJ2K codestream as specified in clause 6.1.

NOTE – Rec. ITU-T T.800 | ISO/IEC 15444-1 specifies that, when displaying the image, all codestreams after the first codestream found in the file are ignored, and that Contiguous Codestream boxes can be found anywhere in the file except before the JP2 Header box.

D.6 Channel Definition box

D.6.1 Single alpha channel

In contrast to the JP2 file format, which supports multiple alpha channels, JPH only supports a single alpha channel.

If the Channel Definition box is present, at most one Typ^i field shall be equal to 1 or 2, and the corresponding $Asoc^i$ field shall be equal to 0.

D.6.2 Multiple channels per colour

In contrast to the JP2 file format, multiple channels can be associated with the same colour.

There may be more than one channel with the same Typ^i and $Asoc^i$ value pair.

EXAMPLE – Multiple channels of the same colour in a Bayer pattern can be described using the same Typ^i and $Asoc^i$ value pair, and but different component registration position, as carried in the optional CRG marker segment.

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

D.6.3 Application-specified colour

This standard defines the additional Typ^i values listed in Table D.4.

Table D.4 – Additional Typ^i field value

Value	Meaning
3	The colour associated with this channel is application-defined.

When Typ^i is equal to 3, the value of $Asoc^i$ shall be between 0 and 2^{16} -1 and is application-defined.

$Asoc^i$ values are application-specific.

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved

Annex E

Media type specifications and registrations

(This annex forms an integral part of this Recommendation | International Standard.)

E.1 General

Many Internet protocols are designed to carry arbitrary labelled content. The mechanism used to label such content is a media type, which is defined in IETF RFC 6838 and consists of a top-level type, a subtype, and in some instances, optional parameters.

The media type specifications of clauses E.2 and E.3 have a matching registration in the Internet Assigned Numbers Authority central registry, as specified in IETF RFC 6838.

E.2 JPH file

E.2.1 General

The `image/jph` media type refers to content that consists of a single JPH file as specified in Annex D.

E.2.2 Registration

```
Type name: image
Subtype name: jph
Required parameters: N/A
Optional parameters: N/A
Encoding considerations: See Section 4.1 of RFC 3745.
Security considerations: See Section 3 of RFC 3745.
Interoperability considerations: N/A
Published specification: Rec. ITU-T T.814 | ISO/IEC 15444-15
Applications: Multimedia and scientific
Fragment identifier considerations: N/A
Restrictions on usage: N/A
Additional information:
   Deprecated alias names for this type: N/A
   Magic number(s): See Section 4.4 of RFC 3745
   File extension(s): jph
   Macintosh File Type Code(s): N/A
Object Identifiers: N/A
Contact name: ISO/IEC JTC 1/SC 29/WG 1 Convenor
Contact email address: sc29-sec@itscj.ipsj.or.jp
Intended usage: COMMON
Change controller: ITU-T & ISO/IEC JTC 1
```

E.3 Single HTJ2K codestream

E.3.1 General

The `image/jphc` media type refers to content that consists of a single HTJ2K codestream, as specified in clause 6.1.

IECNORM.COM : Click to view the full PDF of ISO/IEC 15444-15:2019

© ISO/IEC 2019 – All rights reserved