5 ENCODER

5.1 OVERVIEW

This section specifies the encoding stage of the compressor and the format of a compressed image. Quantities defined in this section are summarized in table E-3 of annex E.

A compressed image consists of a *header* followed by a *body*.

The variable-length header, defined in 5.3, encodes image and compression parameters.

The body, defined in 5.4, losslessly encodes mapped quantizer indices $\delta_{z,y,x}$ from the predictor. If the periodic error limit updating option is used (see 4.8.2.4), then error limit values are also periodically encoded as part of the body.

A user can choose to perform encoding using the *sample-adaptive* entropy coding approach specified in 5.4.3.2, the *hybrid* approach specified in 5.4.3.3, or the *block-adaptive* approach specified in 5.4.3.4.

The sample-adaptive and block-adaptive entropy coding approaches are the same as the ones specified in reference [D2]. They are generally effective for lossless compression, but the ability to use near-lossless compression under the present Recommended Standard tends to yield mapped quantizer indices having a lower-entropy distribution. The hybrid encoding approach tends to provide more effective encoding for lower-entropy distributions. Examples and comparisons can be found in reference [D1].

Under the sample-adaptive entropy coding approach, each mapped quantizer index is encoded using a variable-length binary codeword from a family of codes. Which member of this family is used is adaptively selected based on statistics that are updated after each sample is encoded; separate statistics are maintained for each spectral band, and the compressed image size does not depend on the order in which mapped quantizer indices are encoded.

Like the sample-adaptive coder, the hybrid entropy coding approach uses similar adaptive code selection statistics. It includes codes equivalent to those used by the sample-adaptive encoder, but augmented with an additional 16 variable-to-variable length ‘low-entropy’ codes. A single output codeword from a low-entropy code may encode multiple mapped quantizer indices, allowing lower compressed data rates than can be achieved by the ‘high-entropy’ codes. The order in which mapped quantizer indices are encoded has virtually no impact on compressed image size.

The block-adaptive entropy coding approach relies on the lossless data compressor defined in reference [1]. Under this approach, the sequence of mapped quantizer indices is partitioned into short blocks, and the encoding method used is independently and adaptively selected for each block. Depending on the encoding order, the mapped quantizer indices in a block may be from the same or different spectral bands, and thus the compressed image size depends on the encoding order when this approach is used.

5.2 GENERAL

5.2.1 A compressed image shall consist of a variable-length *header*, defined in 5.3, followed by a variable-length *body*, defined in 5.4.

NOTE – Figure 5-1 depicts the structure of a compressed image.

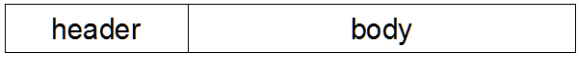

Figure 5-1: Compressed Image Structure

5.2.2 The user-selected *output word size*, measured in bytes, shall be an integer B in the range $1 \leq B \leq 8$.

NOTE – Fill bits are included in the body (as specified in 5.4.3.2.4.4, 5.4.3.4.3.2, and 5.4.3.3.5.4.5) when needed to ensure that the size of the compressed image is a multiple of the output word size.

5.3 HEADER

5.3.1 GENERAL

5.3.1.1 The header of a compressed image shall have the structure specified in table 5-1.

Table 5-1: Header Structure

Part	Status	Size	Reference
Image Metadata	Mandatory	Variable	5.3.2
Predictor Metadata	Mandatory	Variable	5.3.3
Entropy Coder Metadata	Mandatory	Variable	5.3.4

NOTES

1 The length of each header part varies depending on compression options selected by the user.

2 Each header part consists of an integer number of bytes. The header length is not necessarily a multiple of the output word size.

5.3.1.2 Whenever fill bits are included in a header element, fill bits shall be all zeros.

5.3.2 IMAGE METADATA

5.3.2.1 Header

The Image Metadata header part shall have the structure specified in table 5-2.

Table 5-2: Image Metadata Structure

Subpart	Status	Size (Bytes)	Reference
Essential	Mandatory	12	5.3.2.2
Supplementary Information Tables	Optional	Variable	5.3.2.3

5.3.2.2 Essential

The Essential subpart shall have the structure specified in table 5-3.

Table 5-3: Essential Subpart Structure

Field	Width (bits)	Description	Reference
User-Defined Data	8	The user may assign the value of this field arbitrarily, for example, to indicate the value of a user-defined index of the image within a sequence of images.	
X Size	16	The value N_X encoded mod 2^{16} as a 16-bit unsigned binary integer.	3.2
Y Size	16	The value N_Y encoded mod 2^{16} as a 16-bit unsigned binary integer.	3.2
Z Size	16	The value N_Z encoded mod 2^{16} as a 16-bit unsigned binary integer.	3.2
Sample Type	1	'0': image sample values are unsigned integers. '1': image sample values are signed integers.	3.2.1
Reserved	1	This field shall have value '0'.	
Large Dynamic Range Flag	1	'0': dynamic range satisfies $D \leq 16$. '1': dynamic range satisfies $D > 16$.	3.3
Dynamic Range	4	The value D mod 2^4 as a 4-bit unsigned binary integer.	3.3
Sample Encoding Order	1	'0': samples are encoded in band-interleaved order. '1': samples are encoded in BSQ order.	5.4.2
Sub-Frame Interleaving Depth	16	When band-interleaved encoding order is used, this field shall contain the value M encoded mod 2^{16} as a 16-bit unsigned binary integer. When BSQ encoding order is used, this field shall be all 'zeros'.	5.4.2.2
Reserved	2	This field shall have value '00'.	

Field	Width (bits)	Description	Reference
Output Word Size	3	The value B encoded mod 2^3 as a 3-bit unsigned binary integer.	5.2.2
Entropy Coder Type	2	'00': sample-adaptive entropy coder is used. '01': hybrid entropy coder is used. '10': block-adaptive entropy coder is used.	5.4.3
Reserved	1	This field shall have value '0'.	
Quantizer Fidelity Control Method	2	'00': lossless. '01': absolute error limit only. '10': relative error limit only. '11': both absolute and relative error limits.	4.8.2.1
Reserved	2	This field shall contain all 'zeros'.	
Supplementary Information Table Count	4	The value τ, encoded as a 4-bit unsigned integer.	3.5.2.1

5.3.2.3 Supplementary Information Tables

5.3.2.3.1 General

5.3.2.3.1.1 The Supplementary Information Tables subpart shall be present when the number of supplementary information tables, τ, is nonzero and shall be omitted otherwise.

5.3.2.3.1.2 When present, the Supplementary Information Tables subpart shall consist of a sequence of τ Supplementary Information Tables, each having the structure specified in table 5-4.

Table 5-4: Supplementary Information Table Structure

Field	Width (bits)	Description	Reference
Table Type	2	'00': unsigned integer. '01': signed integer. '10': float.	3.5.2.3
Reserved	2	This field shall have value '00'.	
Table Purpose	4	Table purpose value encoded as a 4-bit unsigned integer (see table 3-1).	3.5.2.2
Reserved	1	This field shall have value '0'.	
Table Structure	2	'00': zero-dimensional. '01': one-dimensional. '10': two-dimensional-*zx*. '11': two-dimensional-*yx*.	3.5.2.4
Reserved	1	This field shall have value '0'.	
Supplementary User-Defined Data	4	The user may assign the value of this field arbitrarily.	
Table Data Subblock	(variable)	(See 5.3.2.3.2 below.)	3.5

5.3.2.3.2 Table Data Subblock

5.3.2.3.2.1 A Table Data subblock shall have the structure specified in 5.3.2.3.2.2 when the table type is unsigned or signed integer, and the structure specified in 5.3.2.3.2.3 when the table type is float.

5.3.2.3.2.2 If the Table Type is signed or unsigned integer, then the Table Data subblock shall consist of

a) the value of the table bit depth D_{I} encoded modulo 2^5 as a 5-bit unsigned integer (5 bits);

b) the sequence of table elements (D_{I} bits each):

   1) if the table structure is zero-dimensional, then the single value i is encoded;

   2) if the table structure is one-dimensional, then each i_z is encoded in order of increasing index z;

   3) if the table structure is two-dimensional-zx, then each $i_{z,x}$ is encoded in the order defined by the nesting of loops as follows:

      for $z = 0$ to $N_{\mathrm{Z}} - 1$
         for $x = 0$ to $N_{\mathrm{X}} - 1$
            encode $i_{z,x}$;

   4) if the table structure is two-dimensional-yx, then each $i_{y,x}$ is encoded in the order defined by the nesting of loops as follows:

      for $y = 0$ to $N_{\mathrm{Y}} - 1$
         for $x = 0$ to $N_{\mathrm{X}} - 1$
            encode $i_{y,x}$;

   5) each table element, i, i_z, $i_{z,x}$, or $i_{y,x}$, is encoded as a D_{I}-bit unsigned binary integer, or in two's complement representation for table types unsigned integer and signed integer, respectively; and

c) fill bits appended as needed to reach the next byte boundary.

5.3.2.3.2.3 If the Table Type is float, then the Table Data subblock shall consist of

a) the value of the significand bit depth D_{F} encoded as a 5-bit unsigned integer (5 bits);

b) the value of the exponent bit depth D_{E} encoded mod 2^3 as a 3-bit unsigned integer (3 bits);

c) the value of the exponent bias β encoded as an D_{E}-bit unsigned integer (D_{E} bits);

d) the sequence of table elements ($1 + D_{\mathrm{F}} + D_{\mathrm{E}}$ bits each):

1) if the table structure is zero-dimensional, then the single table element $\{b, \alpha, j\}$ is encoded;

2) if the table structure is one-dimensional, then table elements $\{b_z, \alpha_z, j_z\}$ are encoded in the following order:

```
for z = 0 to N_Z − 1
        encode {b_z, α_z, j_z};
```

3) if the table structure is two-dimensional-*zx*, then table elements $\{b_{z,x}, \alpha_{z,x}, j_{z,x}\}$ are encoded in the order defined by the nesting of loops as follows:

```
for z = 0 to N_Z − 1
        for x = 0 to N_X − 1
                encode {b_{z,x}, α_{z,x}, j_{z,x}};
```

4) if the table structure is two-dimensional-*yx*, then table elements $\{b_{y,x}, \alpha_{y,x}, j_{y,x}\}$ are encoded in the order defined by the nesting of loops as follows:

```
for y = 0 to N_Y − 1
        for x = 0 to N_X − 1
                encode {b_{y,x}, α_{y,x}, j_{y,x}};
```

5) for each table element $\{b, \alpha, j\}$, $\{b_z, \alpha_z, j_z\}$, $\{b_{z,x}, \alpha_{z,x}, j_{z,x}\}$, or $\{b_{y,x}, \alpha_{y,x}, j_{y,x}\}$, the following are encoded:

   i) the value of the sign bit (1 bit);

   ii) the value of the exponent, encoded as a D_E-bit unsigned integer (D_E bits);

   iii) the value of the significand, encoded as an D_F-bit unsigned integer (D_F bits); and

e) fill bits appended as needed to reach the next byte boundary.

5.3.3 PREDICTOR METADATA

5.3.3.1 Header

The Predictor Metadata header part shall have the structure specified in table 5-5.

Table 5-5: Predictor Metadata Structure

Subpart	Status	Size (Bytes)	Reference
Primary	Mandatory	5	5.3.3.2
Weight Tables	Optional	Variable	5.3.3.3
Quantization	Conditional	Variable	5.3.3.4
Sample Representative	Conditional	Variable	5.3.3.5

5.3.3.2 Primary

The Primary subpart shall have the structure specified in table 5-6.

Table 5-6: Primary Structure

Field	Width (bits)	Description	Reference
Reserved	1	This field shall have value '0'.	
Sample Representative Flag	1	'0': Sample Representative subpart is not included in Predictor Metadata header part; sample representatives use $\phi_z = \psi_z = 0$ for all spectral bands z. '1': Sample Representative subpart is included in Predictor Metadata header part.	4.9
Number of Prediction Bands	4	The value P encoded as a 4-bit unsigned binary integer.	4.2
Prediction Mode	1	'0': full prediction mode is used. '1': reduced prediction mode is used.	4.3
Weight Exponent Offset Flag	1	'0': all $\varsigma_z^{(i)}$ and ς_z^* values are zero. '1': some $\varsigma_z^{(i)}$ and ς_z^* values may be nonzero.	4.10.3
Local Sum Type	2	'00': wide neighbor-oriented local sums are used. '01': narrow neighbor-oriented local sums are used. '10': wide column-oriented local sums are used. '11': narrow column-oriented local sums are used.	4.4
Register Size	6	The value R encoded mod 2^6 as a 6-bit unsigned binary integer.	4.7.2
Weight Component Resolution	4	The value $(\Omega - 4)$ encoded as a 4-bit unsigned binary integer.	4.6.1
Weight Update Scaling Exponent Change Interval	4	The value $(\log_2 t_{\text{inc}} - 4)$ encoded as a 4-bit unsigned binary integer.	4.10.2
Weight Update Scaling Exponent Initial Parameter	4	The value $(v_{\min} + 6)$ encoded as a 4-bit unsigned binary integer.	4.10.2
Weight Update Scaling Exponent Final Parameter	4	The value $(v_{\max} + 6)$ encoded as a 4-bit unsigned binary integer.	4.10.2
Weight Exponent Offset Table Flag	1	'0': Weight Exponent Offset Table is not included in Predictor Metadata. '1': Weight Exponent Offset Table is included in Weight Tables subpart of Predictor Metadata.	4.10.3
Weight Initialization Method	1	'0': default weight initialization is used. '1': custom weight initialization is used.	4.6.3
Weight Initialization Table Flag	1	'0': Weight Initialization Table is not included in Predictor Metadata. '1': Weight Initialization Table is included in Weight Tables subpart of Predictor Metadata.	4.6.3
Weight Initialization Resolution	5	When the default weight initialization is used, this field shall have value '00000'. Otherwise, this field shall contain the value Q encoded as a 5-bit unsigned binary integer.	4.6.3

5.3.3.3 Weight Tables

5.3.3.3.1 General

5.3.3.3.1.1 The Weight Tables subpart of the Predictor Metadata header part shall be present if the Weight Initialization Table Flag or Weight Exponent Offset Table Flag is set to '1' and be omitted otherwise.

5.3.3.3.1.2 When present, the Weight Tables subpart shall have the structure specified in table 5-7.

Table 5-7: Weight Tables Subpart Structure

Block	Status	Size	Reference
Weight Initialization Table	Optional	Variable	5.3.3.3.2
Weight Exponent Offset Table	Optional	Variable	5.3.3.3.3

5.3.3.3.2 Weight Initialization Table

5.3.3.3.2.1 The optional Weight Initialization Table may be included only when the custom weight initialization method is selected. The presence of the Weight Initialization Table shall be indicated by setting the Weight Initialization Table Flag field to '1'.

NOTE – Even when the custom weight initialization option is used, the Weight Initialization Table may be omitted. For example, a mission might design a fixed set of custom weight initialization vectors for an instrument to be used throughout a mission and elect to not encode these vectors with each image.

5.3.3.3.2.2 When the Weight Initialization Table is included, the custom weight initialization vectors $\{\Lambda_z\}_{z=0}^{N_Z-1}$ shall be encoded, component-by-component, with each component encoded as a Q-bit signed two's complement binary integer, in the order defined by the nesting of loops as follows:

for $z = 0$ to $N_Z - 1$
    for $j = 0$ to $C_z - 1$
        encode component j of Λ_z.

5.3.3.3.2.3 Fill bits shall be appended to the Weight Initialization Table as needed to reach the next byte boundary.

5.3.3.3.3 Weight Exponent Offset Table

5.3.3.3.3.1 The optional Weight Exponent Offset Table may be included only when the Weight Exponent Offset Flag field is '1'. The presence of the optional Weight Exponent Offset Table shall be indicated by setting the Weight Exponent Offset Table Flag field to '1'.

NOTE – Even when nonzero values of $\varsigma_z^{(i)}$ and ς_z^* are used, the Weight Exponent Offset Table might be omitted. For example, a mission might design a fixed set of custom weight exponent offsets for an instrument to be used throughout a mission and elect to not encode these vectors with each image.

5.3.3.3.3.2 When the Weight Exponent Offset Table is included, the inter-band weight exponent offsets $\varsigma_z^{(i)}$ and inter-band weight exponent offsets ς_z^*, for $z = 0,\ldots, N_Z - 1$ and $i = 1,...,P_z^*$, shall be encoded, component-by-component, with each component encoded as a 4-bit signed two's complement binary integer, in the order defined by the nesting of loops as follows:

for $z = 0$ to $N_Z - 1$
    if full prediction mode is used
        encode ς_z^*
    for $i = 1$ to P_z^*
        encode $\varsigma_z^{(i)}$.

5.3.3.3.3.3 Fill bits shall be appended to the Weight Exponent Offset Table as needed to reach the next byte boundary.

5.3.3.4 Quantization

5.3.3.4.1 General

5.3.3.4.1.1 The Quantization subpart shall be included unless the quantizer fidelity control method is lossless (see 4.8.2.1), in which case it shall be omitted.

5.3.3.4.1.2 When present, the Quantization subpart shall have the structure specified in table 5-8.

Table 5-8: Quantization Subpart Structure

Block	Status	Size (Bytes)	Reference
Error Limit Update Period	Conditional	1	5.3.3.4.2
Absolute Error Limit	Conditional	Variable	5.3.3.4.3
Relative Error Limit	Conditional	Variable	5.3.3.4.4

5.3.3.4.2 Error Limit Update Period

5.3.3.4.2.1 When the Quantization subpart is present, it shall include the Error Limit Update Period block unless BSQ encoding is used, in which case it shall be omitted.

5.3.3.4.2.2 When present, the Error Limit Update Period block shall have the structure specified in table 5-9.

Table 5-9: Error Limit Update Period Block Structure

Field	Width (bits)	Description	Reference
Reserved	1	This field shall have value '0'.	
Periodic Updating Flag	1	'0': periodic error limit updating is not used. '1': periodic error limit updating is used.	4.8.2.4
Reserved	2	This field shall contain all 'zeros'.	
Update Period Exponent	4	When periodic error limit updating is used, this field shall contain the value u encoded as a 4-bit unsigned binary integer. Otherwise, this field shall contain all 'zeros'.	4.8.2.4

5.3.3.4.3 Absolute Error Limit

5.3.3.4.3.1 General

5.3.3.4.3.1.1 The Absolute Error Limit block shall be included when absolute error limits are used and be omitted otherwise.

5.3.3.4.3.1.2 When present, the Absolute Error Limit block shall have the structure specified in table 5-10.

Table 5-10: Absolute Error Limit Block Structure

Field	Width (bits)	Description	Reference
Reserved	1	This field shall have value '0'.	
Absolute Error Limit Assignment Method	1	'0': band-independent absolute error limit assignment. '1': band-dependent absolute error limit assignment.	4.8.2.3.1
Reserved	2	This field shall have value '00'.	
Absolute Error Limit Bit Depth	4	The value D_{A} encoded mod 2^4 as a 4-bit unsigned integer.	4.8.2.2.1
Absolute Error Limit Values Subblock (conditional)	(variable)	(See 5.3.3.4.3.2 below.)	4.8.2

5.3.3.4.3.2 Absolute Error Limit Values Subblock

5.3.3.4.3.2.1 When the Absolute Error Limit block is present, it shall include the Absolute Error Limit Values Subblock unless periodic error limit updating is used, in which case it shall be omitted.

5.3.3.4.3.2.2 If band-independent absolute error limits are used, then the Absolute Error Limit Values Subblock consists of the value A^* encoded as a D_A-bit unsigned binary integer, followed by fill bits as needed to reach the next byte boundary.

5.3.3.4.3.2.3 If band-dependent absolute error limits are used, then the Absolute Error Limit Values Subblock shall consist of (a) the sequence of a_z values, in order of increasing band index z, each encoded as a D_A-bit unsigned binary integer, followed by (b) fill bits as needed to reach the next byte boundary.

5.3.3.4.4 Relative Error Limit

5.3.3.4.4.1 General

5.3.3.4.4.1.1 The Relative Error Limit block shall be included when relative error limits are used and shall be omitted otherwise.

5.3.3.4.4.1.2 When present, the Relative Error Limit block shall have the structure specified in table 5-11.

Table 5-11: Relative Error Limit Block Structure

Field	Width (bits)	Description	Reference
Reserved	1	This field shall have value ‘0’.	
Relative Error Limit Assignment Method	1	‘0’: band-independent relative error limit assignment. ‘1’: band-dependent relative error limit assignment.	4.8.2.3.2
Reserved	2	This field shall have value ‘00’.	
Relative Error Limit Bit Depth	4	The value D_R encoded mod 2^4 as a 4-bit unsigned integer.	4.8.2.2.2
Relative Error Limit Values Subblock (conditional)	(variable)	(See 5.3.3.4.4.2 below.)	

5.3.3.4.4.2 Relative Error Limit Values Subblock

5.3.3.4.4.2.1 When the Relative Error Limit block is present, it shall include the Relative Error Limit Values Subblock unless periodic error limit updating is used, in which case it shall be omitted.

5.3.3.4.4.2.2 If band-independent relative error limits are used, then the Relative Error Limit Values Subblock shall consist of the value R^* encoded as a D_R-bit unsigned binary integer, followed by fill bits as needed to reach the next byte boundary.

5.3.3.4.4.2.3 If band-dependent relative error limits are used, then the Relative Error Limit Values Subblock shall consist of (a) the sequence of r_z values, in order of increasing band index z, each encoded as a D_R-bit unsigned binary integer, followed by (b) fill bits as needed to reach the next byte boundary.

5.3.3.5 Sample Representative

5.3.3.5.1 General

5.3.3.5.1.1 The Sample Representative subpart may only be included when $\Theta > 0$. The inclusion of the Sample Representative subpart shall be indicated by setting the Sample Representative Flag field bit to '1'.

5.3.3.5.1.2 When present, the Sample Representative subpart shall have the structure specified in table 5-12.

Table 5-12: Sample Representative Subpart Structure

Field	Width (bits)	Description	Reference
Reserved	5	This field shall contain all 'zeros'.	
Sample Representative Resolution	3	Value of Θ encoded as a 3-bit unsigned binary integer.	4.9.1
Reserved	1	This field shall have value '0'.	
Band-Varying Damping Flag	1	'0': all bands use the same value of ϕ_z. '1': the value ϕ_z of may vary from band to band.	4.9.1
Damping Table Flag	1	'0': the Damping Table subblock is not included in the Sample Representative subpart. '1': the Damping Table subblock is included in the Sample Representative subpart.	4.9.1
Reserved	1	This field shall have value '0'.	

Field	Width (bits)	Description	Reference
Fixed Damping Value	4	If the Band-Varying Damping Flag field is '0', then this field encodes the value of ϕ_z to use for all bands as a 4-bit unsigned integer. Otherwise, this field shall be all 'zeros'.	4.9.1
Reserved	1	This field shall have value '0'.	
Band-Varying Offset Flag	1	'0': all bands use the same value of ψ_z. '1': the value of ψ_z may vary from band to band.	4.9.1
Offset Table Flag	1	'0': the Offset Table subblock is not included in the Sample Representative subpart. '1': the Offset Table subblock is included in the Sample Representative subpart.	4.9.1
Reserved	1	This field shall have value '0'.	
Fixed Offset Value	4	If the Band-Varying Offset Field Flag field is '0', then this field encodes the value of ψ_z to use for all bands as a 4-bit unsigned integer. Otherwise, this field shall be all 'zeros'.	4.9.1
Damping Table Subblock (optional)	(variable)	(See 5.3.3.5.2 below.)	4.9.1
Offset Table Subblock (optional)	(variable)	(See 5.3.3.5.3 below.)	4.9.1

5.3.3.5.2 Damping Table Subblock

5.3.3.5.2.1 The optional Damping Table Subblock may only be included when the Band-Varying Damping Flag field is '1'. The inclusion of the Damping Table Subblock shall be indicated by setting the Damping Table Flag field to '1'.

NOTE – Even when the damping value ϕ_z varies from band to band, the Damping Table Subblock might be omitted. For example, a mission might design a fixed set of damping values to be used throughout a mission and elect to not encode these values with each image.

5.3.3.5.2.2 When present, the Damping Table Subblock shall consist of (a) the sequence of ϕ_z values, in order of increasing band index z, each encoded as a Θ-bit unsigned binary integer, followed by (b) fill bits as needed to reach the next byte boundary.

5.3.3.5.3 Offset Table Subblock

5.3.3.5.3.1 The optional Offset Table Subblock may only be included when the Band-Varying Offset Flag field is '1'. The inclusion of the Offset Table Subblock shall be indicated by setting the Offset Table Flag field to '1'.

NOTE – Even when the offset value ψ_z varies from band to band, the Offset Table Subblock might be omitted. For example, a mission might design a fixed set of offset values to be used throughout a mission and elect to not encode these values with each image.

5.3.3.5.3.2 When present, the Offset Table Subblock shall consist of (a) the sequence of ψ_z values, in order of increasing band index z, each encoded as a Θ-bit unsigned binary integer, followed by (b) fill bits as needed to reach the next byte boundary.

5.3.4 ENTROPY CODER METADATA

5.3.4.1 General

The Entropy Coder Metadata header part shall follow the structure defined in 5.3.4.2 if the sample-adaptive entropy coder is used, the structure defined in 5.3.4.3 if the hybrid entropy coder is used, or the structure defined in 5.3.4.4 if the block-adaptive entropy coder is used.

5.3.4.2 Sample-Adaptive Entropy Coder

5.3.4.2.1 Header

When the sample-adaptive entropy coder is used, the Entropy Coder Metadata header part shall have the structure specified in table 5-13.

Table 5-13: Entropy Coder Metadata Structure When Sample Adaptive Entropy Coder Is Used

Field	Width (bits)	Description	Reference
Unary Length Limit	5	The value $U_{\max}$ encoded mod 2^5 as a 5-bit unsigned binary integer.	5.4.3.2.2
Rescaling Counter Size	3	The value $(\gamma^* - 4)$ encoded as a 3-bit unsigned binary integer.	5.4.3.2.3.4
Initial Count Exponent	3	The value γ_0 encoded mod 2^3 as a 3-bit unsigned binary integer.	5.4.3.2.3.2
Accumulator Initialization Constant	4	When an accumulator initialization constant K is specified, this field encodes the value of K as a 4-bit unsigned binary integer. Otherwise, this field shall be all 'ones'.	5.4.3.2.3.3
Accumulator Initialization Table Flag	1	'0': Accumulator Initialization Table is not included in Entropy Coder Metadata. '1': Accumulator Initialization Table is included in Entropy Coder Metadata.	5.4.3.2.3.3
Accumulator Initialization Table (Optional)	(variable)	(See 5.3.4.2.2 below.)	5.4.3.2.3.3

5.3.4.2.2 Accumulator Initialization Table

5.3.4.2.2.1 The optional Accumulator Initialization Table may be included when an accumulator initialization constant is not specified. The presence of an accumulator initialization table shall be indicated by setting the Accumulator Initialization Table Flag field to '1'.

NOTE – Even when an accumulator initialization constant is not used, the Accumulator Initialization Table may be omitted. For example, a mission might design a fixed set of accumulator initialization values to be used throughout a mission and elect to not encode these values with each image.

5.3.4.2.2.2 The Accumulator Initialization Table shall consist of the concatenated sequence of k_z'' values, $k_0'', k_1'', \ldots, k_{N_Z-1}''$ (defined in 5.4.3.2.3.3), each encoded as a 4-bit binary unsigned integer.

5.3.4.2.2.3 Fill bits shall be appended to the Accumulator Initialization Table as needed to reach the next byte boundary.

5.3.4.3 Hybrid Entropy Coder

When the hybrid entropy coder is used, the Entropy Coder Metadata header part shall have the structure specified in table 5-14.

Table 5-14: Entropy Coder Metadata Structure When Hybrid Entropy Coder Is Used

Field	Width (bits)	Description	Reference
Unary Length Limit	5	The value $U_{\max}$ encoded mod 2^5 as a 5-bit unsigned binary integer.	5.4.3.3.3.2.2
Rescaling Counter Size	3	The value $(\gamma^* - 4)$ encoded as a 3-bit unsigned binary integer.	5.4.3.3.4.4
Initial Count Exponent	3	The value γ_0 encoded mod 2^3 as a 3-bit unsigned binary integer.	5.4.3.3.4.2
Reserved	5	This field shall have value '00000'.	

5.3.4.4 Block-Adaptive Entropy Coder

When the block-adaptive entropy coder is used, the Entropy Coder Metadata header part shall have the structure specified in table 5-15.

Table 5-15: Entropy Coder Metadata Structure When Block Adaptive Entropy Coder Is Used

Field	Width (bits)	Description	Reference
Reserved	1	This field shall have value '0'.	
Block Size	2	'00': Block size $J = 8$. '01': Block size $J = 16$. '10': Block size $J = 32$. '11': Block size $J = 64$.	5.4.3.4.2.4
Restricted Code Options Flag	1	This field shall have value '1' when $D \leq 4$ and the Restricted set of code options (as defined in subsection 5.1.2 of reference [1]) are used. Otherwise, this field shall have value '0'.	
Reference Sample Interval	12	Value of r encoded mod 2^{12} as a 12-bit unsigned binary integer.	5.4.3.4.2.5

5.4 BODY

5.4.1 OVERVIEW

The *entropy coder input sequence* consists of the mapped quantizer indices, and, when periodic error limit updating is used (see 4.8.2.4), quantizer error limit values. This input sequence is arranged in one of the allowed orders specified in 5.4.2. The compressed image body losslessly encodes this sequence using one of the three entropy coding methods specified in 5.4.3.

5.4.2 INPUT ORDER

5.4.2.1 General

The entropy coder input sequence shall be arranged in Band-Interleaved (BI) order, as defined in 5.4.2.2, or BSQ order, as defined in 5.4.2.3.

NOTES

1 The input order specifies the order in which the entropy coder input sequence values are input to the entropy coder.

2 The commonly used Band-Interleaved-by-Pixel (BIP) and Band-Interleaved-by-Line (BIL) orders are each special cases of the more general BI encoding order.

3 The entropy coder input sequence order does not necessarily correspond to the order in which samples are produced by an imaging instrument or processed by a predictor implementation.

5.4.2.2 Band-Interleaved Order

5.4.2.2.1 The user-specified *sub-frame interleaving depth M* shall be an integer in the range $1 \leq M \leq N_Z$.

5.4.2.2.2 Under BI input order, the entropy coder input sequence order is defined by the nesting of sample index loops as follows:

for $y = 0$ to $N_Y - 1$
    if $y \bmod 2^u = 0$ and periodic error limit updating is used
        if absolute error limits are used
            if absolute error limits are band-independent
                input A^* to the entropy coder
            else
                input $a_0, a_1, ..., a_{N_Z-1}$ to the entropy coder

```
                    if relative error limits are used
                        if relative error limits are band-independent
                            input R* to the entropy coder
                        else
                            input r_0, r_1, ..., r_{N_Z-1} to the entropy coder
    for i = 0 to ⌈N_Z / M⌉ − 1
        for x = 0 to N_X − 1
            for z = iM to min{(i+1)M − 1, N_Z − 1}
                input δ_{z,y,x} to the entropy coder.
```

NOTES

1 Under BI encoding order, when $M = 1$, the input order corresponds to BIL, and when $M = N_Z$ the input order corresponds to BIP.

2 When periodic error limit updates are not used, error limit values are not part of the entropy coder input sequence and instead are encoded in the header as specified in 5.3.3.4.

5.4.2.3 Band-Sequential Order

Under BSQ input order, the entropy coder input sequence order is defined by the nesting of sample index loops as follows:

```
for z = 0 to N_Z − 1
    for y = 0 to N_Y − 1
        for x = 0 to N_X − 1
            input δ_{z,y,x} to the entropy coder.
```

NOTE – As specified in 4.8.2.4, periodic error limit updates are not permitted when BSQ encoding order is used.

5.4.3 ENTROPY CODING METHOD

5.4.3.1 General

The entropy coder input sequence shall be encoded using either the sample-adaptive entropy coding approach specified in 5.4.3.2, the hybrid entropy coding approach specified in 5.4.3.3, or the block-adaptive entropy coding approach specified in 5.4.3.4.

5.4.3.2 Sample-Adaptive Entropy Coder

5.4.3.2.1 General

Under the sample-adaptive entropy coding option, each mapped quantizer index $\delta_z(t)$ shall be encoded using a variable-length binary codeword.

NOTE – The family of variable-length codes used is defined in 5.4.3.2.2, and the adaptive code selection statistics used to select the codeword for each mapped quantizer index are specified in 5.4.3.2.3. The procedure for selecting the codeword for each mapped quantizer index and encoding error limit values is specified in 5.4.3.2.4.

5.4.3.2.2 Length-Limited Golomb-Power-of-2 Codewords

5.4.3.2.2.1 The length-limited Golomb-power-of-2 (GPO2) codeword for unsigned integer j and unsigned integer code index k, denoted $\Re_k(j)$, is a variable-length binary codeword defined as follows:

a) if $\lfloor j/2^k \rfloor < U_{\max}$ then $\Re_k(j)$ consists of $\lfloor j/2^k \rfloor$ 'zeros', followed by a 'one', followed by the k least significant bits of the binary representation of j;

b) otherwise, $\Re_k(j)$ consists of $U_{\max}$ 'zeros' followed by the D-bit binary representation of j.

5.4.3.2.2.2 The user-specified unary length limit $U_{\max}$ shall be an integer in the range $8 \leq U_{\max} \leq 32$.

NOTE – The definition ensures that each codeword $\Re_k(j)$ is not longer than $U_{\max} + D$ bits.

5.4.3.2.3 Adaptive Code Selection Statistics

5.4.3.2.3.1 The adaptive code selection statistics shall consist of an *accumulator* $\Sigma_z(t)$ and a *counter* $\Gamma(t)$ that are adaptively updated during the encoding process.

NOTE – The ratio $\Sigma_z(t) / \Gamma(t)$ provides an estimate of the mean mapped quantizer index value in the spectral band. This ratio determines the variable-length code used to encode $\delta_z(t)$.

5.4.3.2.3.2 The initial counter value $\Gamma(1)$ shall be equal to

$$\Gamma(1) = 2^{\gamma_0}, \tag{57}$$

where the user-specified value of the initial count exponent γ_0 shall be an integer in the range $1 \le \gamma_0 \le 8$.

5.4.3.2.3.3 For each spectral band z, the initial accumulator value $\Sigma_z(1)$ shall be equal to

$$\Sigma_z(1) = \left\lfloor \frac{1}{2^7}\left(3 \cdot 2^{k_z'+6} - 49\right)\Gamma(1) \right\rfloor, \tag{58}$$

where

$$k_z' = \begin{cases} k_z'', & k_z'' \le 30 - D \\ 2k_z'' + D - 30, & k_z'' > 30 - D \end{cases}, \tag{59}$$

and the user-selected value k_z'' shall be an integer in the range $0 \le k_z''' \le \min(D-2, 14)$. An *accumulator initialization constant* K may be specified, with $0 \le K \le \min(D-2, 14)$, in which case $k_z'' = K$ for all z.

NOTE – This calculation ensures that initial value of encoding parameter $k_z(t)$ computed for spectral band z (see 5.4.3.2.4.3) will be equal to k_z'.

5.4.3.2.3.4 For $t > 1$, the value of the accumulator for spectral band z is defined as

$$\Sigma_z(t) = \begin{cases} \Sigma_z(t-1) + \delta_z(t-1), & \Gamma(t-1) < 2^{\gamma^*} - 1 \\ \left\lfloor \dfrac{\Sigma_z(t-1) + \delta_z(t-1) + 1}{2} \right\rfloor, & \Gamma(t-1) = 2^{\gamma^*} - 1 \end{cases}, \tag{60}$$

and the value of the counter is defined as

$$\Gamma(t) = \begin{cases} \Gamma(t-1) + 1, & \Gamma(t-1) < 2^{\gamma^*} - 1 \\ \left\lfloor \dfrac{\Gamma(t-1) + 1}{2} \right\rfloor, & \Gamma(t-1) = 2^{\gamma^*} - 1 \end{cases}. \tag{61}$$

The interval at which the counter $\Gamma(t)$ and the accumulator $\Sigma_z(t)$ are rescaled is controlled by the user-defined rescaling counter size parameter γ^*, which shall be an integer in the range $\max\{4, \gamma_0 + 1\} \le \gamma^* \le 11$.

5.4.3.2.4 Coding Procedure

5.4.3.2.4.1 Each absolute error limit value shall be encoded as a D_A-bit unsigned binary integer, and each relative error limit value shall be encoded as a D_R-bit unsigned binary integer.

NOTE – The adaptive code selection statistics are unaffected by the encoding of error limit values.

5.4.3.2.4.2 The first mapped quantizer index in each spectral band z shall be uncoded; that is, the codeword for $\delta_z(0)$ is simply the D-bit unsigned binary integer representation of $\delta_z(0)$.

5.4.3.2.4.3 For $t > 0$, the codeword for the mapped quantizer index $\delta_z(t)$ is $\mathfrak{R}_{k_z(t)}(\delta_z(t))$, where $k_z(t) = 0$ if $2\Gamma(t) > \Sigma_z(t) + \left\lfloor \frac{49}{2^7}\Gamma(t) \right\rfloor$; otherwise, $k_z(t)$ is the largest positive integer $k_z(t) \leq D - 2$, such that

$$\Gamma(t)2^{k_z(t)} \leq \Sigma_z(t) + \left\lfloor \frac{49}{2^7}\Gamma(t) \right\rfloor. \tag{62}$$

5.4.3.2.4.4 Following the last codeword in the compressed image, fill bits shall be appended as needed to reach the next output word boundary, so that the compressed image size is a multiple of the output word size. Fill bits shall be all ‘zeros’.

5.4.3.3 Hybrid Entropy Coder

5.4.3.3.1 Overview

Under the hybrid entropy coding option, adaptive code selection statistics are used to assign each mapped quantizer index to either a ‘high-entropy’ or ‘low-entropy’ coding method. Each high-entropy mapped quantizer index is encoded using a variable-length binary codeword from a family of codes. For each low-entropy mapped quantizer index, one of 16 variable-to-variable length codes is used. A single output codeword from a low-entropy code can encode multiple input-mapped quantizer indices, which allows lower compressed data rates than can be achieved by the high-entropy codes. Each high-entropy mapped quantizer index immediately produces an output codeword that is written to the compressed bitstream, while each low-entropy code waits until enough data has arrived to determine the next output codeword.

The decoder can accommodate the varying latency between the arrival of a low-entropy mapped quantizer index and its ultimate encoding by decoding the compressed image body in reverse order. This is possible because (1) the output codewords from the high- and low-entropy codes are suffix-free rather than prefix-free, (2) the compressed image body ends with a compressed image ‘tail’ (see 5.4.3.3.5.4) that encodes the final state of each low-entropy code and the final high-resolution accumulator value for each band, and (3) each

time the adaptive code selection statistics are rescaled (see 5.4.3.3.4.4), an additional bit is output (see 5.4.3.3.5.1.2) so that the decoder can invert this rescaling operation. Because decoding proceeds in reverse, users need to provide a mechanism by which the decoder can locate the end of the compressed image body (see 2.4).

5.4.3.3.2 General

The high-entropy and low-entropy encoding methods used to encode mapped quantizer indices are specified in 5.4.3.3.3. The coding method selection depends on the adaptive code selection statistics specified in 5.4.3.3.4. Following the processing of the entropy coder input sequence using the procedure specified in 5.4.3.3.5, the compressed image body concludes with the compressed image tail, specified in 5.4.3.3.5.4.

5.4.3.3.3 Encoding Methods

5.4.3.3.3.1 Overview

A mapped quantizer index is encoded using one of several reversed length-limited GPO2 codes specified in 5.4.3.3.3.2, or using one of 16 low-entropy codes specified in 5.4.3.3.3.3.

5.4.3.3.3.2 Reversed Length-Limited Golomb-Power-of-2 Codewords

5.4.3.3.3.2.1 The reversed length-limited GPO2 codeword for unsigned integer j and unsigned integer code index k, denoted $\mathfrak{R}'_k(j)$, is a variable-length binary codeword defined as follows:

a) if $\lfloor j/2^k \rfloor < U_{\max}$ then $\mathfrak{R}'_k(j)$ consists of the k least significant bits of the binary representation of j, followed by a 'one', followed by $\lfloor j/2^k \rfloor$ 'zeros';

b) otherwise, $\mathfrak{R}'_k(j)$ consists of the D-bit binary representation of j followed by $U_{\max}$ 'zeros'.

NOTE – The codewords $\mathfrak{R}'_k(j)$ and $\mathfrak{R}_k(j)$ are equivalent, but with the bits arranged in a different order. Also, $\mathfrak{R}'_k(j)$ is not in general the reverse of $\mathfrak{R}_k(j)$.

5.4.3.3.3.2.2 The user-specified unary length limit $U_{\max}$ shall be an integer in the range $8 \leq U_{\max} \leq 32$.

5.4.3.3.3.3 Low-Entropy Codes

5.4.3.3.3.3.1 The low-entropy codes are a set of 16 non-binary-input, binary-output, variable-to-variable length codes. Each low-entropy shall consist of

a) a threshold value T_i and input symbol limit L_i, values for both of which shall be those given in table 5-16;

b) a code defined by a prefix-free set of non-binary variable-length *input codewords* with a mapping onto a set of variable-length binary *output codewords*; and

c) a flush table that gives a mapping from the set of all proper prefixes of input codewords onto a set of output flush words.

NOTE – The code table and flush table for each low-entropy code are specified in annex B.

Table 5-16: Low-Entropy Code Input Symbol Limit and Threshold

Code Index, *i*	Input Symbol Limit, L_i	Threshold, T_i
0	12	303336
1	10	225404
2	8	166979
3	6	128672
4	6	95597
5	4	69670
6	4	50678
7	4	34898
8	2	23331
9	2	14935
10	2	9282
11	2	5510
12	2	3195
13	2	1928
14	2	1112
15	0	408

5.4.3.3.3.3.2 During encoding, each low-entropy code has an *active prefix*, which is a sequence of input symbols. Initially, the active prefix for each low-entropy code shall be equal to the null (empty) sequence.

5.4.3.3.4 Adaptive Code Selection Statistics

5.4.3.3.4.1 The adaptive code selection statistics for the hybrid entropy coder shall consist of a *high-resolution accumulator* $\tilde{\Sigma}_z(t)$ and a *counter* $\Gamma(t)$, which are adaptively updated during the encoding process.

NOTE – The ratio $\tilde{\Sigma}_z(t)/\Gamma(t)$ provides a scaled estimate of the mean mapped quantizer index value in the spectral band. This ratio determines how $\delta_z(t)$ is encoded.

5.4.3.3.4.2 The initial counter value $\Gamma(0)$ shall be equal to

$$\Gamma(0) = 2^{\gamma_0}, \tag{63}$$

where the user-specified value of the initial count exponent γ_0 shall be an integer in the range $1 \leq \gamma_0 \leq 8$.

5.4.3.3.4.3 For each spectral band z, the initial high-resolution accumulator value $\tilde{\Sigma}_z(0)$ shall be a user-specified integer in the range $0 \leq \tilde{\Sigma}_z(0) < 2^{D+\gamma_0}$.

NOTES

1 The value of $\tilde{\Sigma}_z(0)$ is not directly encoded in the header or bitstream.

2 If an estimate $\hat{\delta}_z$ (such an estimate might arise from a preceding compressed image) of the mean mapped quantizer index for spectral band z is available, then a reasonable rule-of-thumb is to initialize $\tilde{\Sigma}_z(0)$ to be approximately equal to $4\Gamma(0)\hat{\delta}_z$.

5.4.3.3.4.4 For $t \geq 1$, the value of the high-resolution accumulator for spectral band z is defined as

$$\tilde{\Sigma}_z(t) = \begin{cases} \tilde{\Sigma}_z(t-1) + 4\delta_z(t), & \Gamma(t-1) < 2^{\gamma^*} - 1 \\ \left\lfloor \dfrac{\tilde{\Sigma}_z(t-1) + 4\delta_z(t) + 1}{2} \right\rfloor, & \Gamma(t-1) = 2^{\gamma^*} - 1 \end{cases}, \tag{64}$$

and the value of the counter is defined as

$$\Gamma(t) = \begin{cases} \Gamma(t-1)+1, & \Gamma(t-1) < 2^{\gamma^*} - 1 \\ \left\lfloor \dfrac{\Gamma(t-1)+1}{2} \right\rfloor, & \Gamma(t-1) = 2^{\gamma^*} - 1 \end{cases}. \tag{65}$$

5.4.3.3.4.5 The interval at which the counter $\Gamma(t)$ and the high-resolution accumulator $\tilde{\Sigma}_z(t)$ are rescaled is controlled by the user-defined rescaling counter size parameter γ^*, which shall be an integer in the range $\max\{4, \gamma_0 + 1\} \leq \gamma^* \leq 11$.

5.4.3.3.5 Coding Procedure

5.4.3.3.5.1 General

5.4.3.3.5.1.1 Each absolute error limit value shall be encoded as a D_A-bit unsigned binary integer, and each relative error limit value shall be encoded as a D_R-bit unsigned binary integer.

NOTE – The adaptive code selection statistics are unaffected by the encoding of error limit values.

5.4.3.3.5.1.2 When $\Gamma(t-1) = 2^{\gamma^*} - 1$ (i.e., when code selection statistics are rescaled, as described in 5.4.3.3.4.4), the least-significant bit of $\tilde{\Sigma}_z(t-1)$ shall be encoded in the bitstream (i.e., a single '1' bit when this quantity is odd and a '0' bit when it is even) immediately before any bits output as a result of the processing steps for $\delta_z(t)$ specified below.

NOTE – This bit allows the decoder to reconstruct the sequence of high-resolution accumulator values.

5.4.3.3.5.1.3 The first mapped quantizer index in each spectral band z shall be uncoded; that is, the D-bit unsigned binary integer representation of $\delta_z(0)$ is output to the compressed bitstream.

5.4.3.3.5.1.4 For $t > 0$, if $\tilde{\Sigma}_z(t) \cdot 2^{14} \geq T_0 \cdot \Gamma(t)$, then $\delta_z(t)$ is said to be a 'high-entropy' mapped quantized index and shall be encoded using a reversed length-limited GPO2 code as described below in 5.4.3.3.5.2. Otherwise, $\delta_z(t)$ is said to be a 'low-entropy' mapped quantized index and shall be processed as described below in 5.4.3.3.5.3.

NOTE – When D=2, the condition for using the high-entropy coding method is never met; all mapped quantizer indices are low-entropy.

5.4.3.3.5.2 High-Entropy Processing

If $\delta_z(t)$ is a high-entropy mapped quantizer index, then it shall be encoded by appending codeword $\mathfrak{R}'_{k_z(t)}(\delta_z(t))$ to the compressed bitstream, where $k_z(t)$ is the largest positive integer $k_z(t) \leq \max\{D-2, 2\}$, such that

$$\Gamma(t)2^{k_z(t)+2} \leq \tilde{\Sigma}_z(t) + \left\lfloor \frac{49}{2^5}\Gamma(t) \right\rfloor. \tag{66}$$

NOTE – For high-entropy samples, it can be shown that $k_z(t) \geq 2$.

5.4.3.3.5.3 Low-Entropy Processing

5.4.3.3.5.3.1 If $\delta_z(t)$ is a low-entropy mapped quantizer index, then it shall be encoded using the low-entropy code with largest code index i satisfying $\tilde{\Sigma}_z(t) \cdot 2^{14} < \Gamma(t) \cdot T_i$.

5.4.3.3.5.3.2 The *input symbol* to the low-entropy code is

$$\iota_z(t) = \begin{cases} \delta_z(t), & \delta_z(t) \leq L_i \\ \mathrm{X}, & \delta_z(t) > L_i \end{cases}, \tag{67}$$

where L_i is the input symbol limit for the code, and 'X' denotes the 'escape' symbol.

5.4.3.3.5.3.3 If $\iota_z(t) = \mathrm{X}$, then the residual value $\delta_z(t) - L_i - 1$ shall be encoded by appending codeword $\mathfrak{R}'_0(\delta_z(t) - L_i - 1)$ to the compressed bitstream.

5.4.3.3.5.3.4 The active prefix for the i^{th} low-entropy code is updated by appending the input symbol $\iota_z(t)$ to that active prefix.

5.4.3.3.5.3.5 If after updating the active prefix it is equal to a complete input codeword, as specified in the code table for that code, then

a) the corresponding output codeword listed in the table shall be appended to the compressed bitstream; and

b) the active prefix for the low-entropy code shall be reset to the null sequence.

NOTE – The low-entropy code designs ensure that the active prefix is always equal to a complete input codeword whenever the input symbol is the escape symbol.

5.4.3.3.5.4 Compressed Image Tail

5.4.3.3.5.4.1 Following the processing of the entropy coder input sequence as specified in 5.4.3.3.5.1.1–5.4.3.3.5.1.4, the compressed image tail shall be produced by using the low-entropy code flush tables to encode the active prefix of each low-entropy code as described in 5.4.3.3.5.4.2, encoding the final high-resolution accumulator value in each band as described in 5.4.3.3.5.4.3, appending an additional '1' bit as described in 5.4.3.3.5.4.4, and appending fill bits (if needed) as described in 5.4.3.3.5.4.5.

5.4.3.3.5.4.2 For each low-entropy code, in order of increasing code index, the active prefix for the low-entropy code shall be encoded by writing the corresponding flush codeword (given in the code's flush table in annex B) to the compressed bitstream.

NOTE – The flush code tables define an output codeword for each possible active prefix, including the null sequence. Thus a flush codeword is output for each of the 16 low-entropy codes.

5.4.3.3.5.4.3 For each spectral band z, in order of increasing band index, the final high-resolution accumulator value $\tilde{\Sigma}_z(N_X \cdot N_Y - 1)$ shall be encoded directly as an unsigned integer using $2 + D + \gamma^*$ bits.

5.4.3.3.5.4.4 Following the encoding of final high-resolution accumulator values, a single '1' bit shall be appended.

NOTE – This '1' bit allows the decoder to identify fill bits encoded in 5.4.3.3.5.4.5.

5.4.3.3.5.4.5 Fill bits shall be appended as needed to reach the next output word boundary, so that the compressed image size is a multiple of the output word size. Fill bits shall be all 'zeros'.

5.4.3.4 Block-Adaptive Entropy Coder

5.4.3.4.1 General

When the block-adaptive entropy coding method is used, the entropy coder input sequence shall be encoded using the adaptive entropy coder specified in reference [1].

5.4.3.4.2 Parameters and Options

5.4.3.4.2.1 When the block-adaptive entropy coding method is used, the following options and parameters shall apply.

5.4.3.4.2.2 The preprocessor function defined in section 4 of reference [1] shall not be used. The option to bypass the preprocessor shall be used.

5.4.3.4.2.3 The *resolution* parameter, *n*, defined in subsection 3.1 of reference [1], shall be equal to the image dynamic range *D*.

5.4.3.4.2.4 The *block size* parameter, *J*, defined in subsection 3.1 of reference [1], shall be equal to 8, 16, 32, or 64.

5.4.3.4.2.5 The *reference sample interval* parameter, *r*, defined in subsection 4.3 of reference [1], shall be a positive integer not larger than 4096.

NOTE – Because the preprocessor is bypassed, reference samples are not included in the compressed image body. The reference sample interval serves only to define an interval of input data sample blocks that will be further segmented in the 'zero-block' encoding option defined in reference [1].

5.4.3.4.2.6 Either the Basic or Restricted set of code options, as defined in subsection 5.1.2 of reference [1], may be used.

5.4.3.4.2.7 The input to the adaptive entropy coder specified in reference [1] shall be the entropy coder input sequence, as specified in 5.4.2, with 'zeros' appended as needed so that the length is a multiple of *J*.

5.4.3.4.3 Body

5.4.3.4.3.1 The compressed image body shall consist of the concatenation of the Coded Data Sets (CDSes), defined in subsection 5.1.4 of reference [1], produced by the encoder.

5.4.3.4.3.2 Fill bits shall be appended after the last CDS as needed to reach the next output word boundary, so that the compressed image size is a multiple of the output word size. Fill bits shall be all 'zeros'. Fill bits shall not be inserted between CDSes.