Department of Justice

Federal Bureau of Investigation

CRIMINAL JUSTICE INFORMATION SERVICES (CJIS)

WSQ GRAY-SCALE FINGERPRINT IMAGE COMPRESSION SPECIFICATION

Version 3.1

October 4, 2010

Prepared by:

Criminal Justice Information Services Division
Federal Bureau of Investigation
1000 Custer Hollow Road
Clarksburg, WV 26306

Change History

A. IAFIS-IC-0110(V2), February 16, 1993. This version is the IAFIS CM Group (CMG), as defined by the IAFIS CM Plan (April, 1997), initial baseline. Prior versions of this specification were maintained outside the purview of the IAFIS CMG.

B. IAFIS-IC-0110(V3), December 19, 1997 - This version incorporates RFC 0895 – Replaced Annex AA: Procedures for Determining Compliance. Replaced Part 3: FBI Parameter Settings, Encoder Number One. Corrected errors in Figure A.2, Figure B.3, and in para B.2.4.3.

C. IAFIS-IC-0110(V3.1), October 1, 2010 –This version incorporates a new variance estimate in section 3.1: Subband variance computation of Part 3: FBI Parameter Settings, Encoder Number One. This change will be called "Encoder Number Two" and requires that frame header variable "Ev" (page 20) be set to 0x02.

Record of Current Changes

RFC	REVISION	SECTION	DESCRIPTION OF CHANGE

Table of Contents

List of Tables

List of Figures

WSQ Gray-scale Fingerprint Image Compression Specification

Part 1 - Requirements and guidelines

1 Scope

This Specification is applicable to continuous-tone gray-scale digital fingerprint images. It is intended for use in conjunction with the proposed revision to ANSI/NBS-CLS 1-1993 American National Standard Data Format For The Interchange Of Fingerprint Information [4].

This specification

- specifies a class of encoders for converting source fingerprint image data to compressed image data;

- specifies a decoder process for converting compressed image data to reconstructed fingerprint image data;

- specifies coded representations for compressed image data.

2 Introduction

The purpose of this Section is to give an informative overview of the elements specified in this Specification. Another purpose is to introduce many of the terms which are defined in Section 3. These terms are printed in *italics* upon first usage in this Section.

2.1 Wavelet Scalar Quantization (WSQ) compression

The WSQ class of encoders involves a decomposition of the fingerprint image into a number of subbands, each of which represents information in a particular frequency band. The subband decomposition is achieved by a *discrete wavelet transformation* of the fingerprint image.

Each of the subbands is then *quantized* using values from a *quantization table*. No default values for quantization tables are given in this Specification.

The quantized coefficients are then passed to a *Huffman encoding* procedure which compresses the data. *Huffman table* specifications must be provided to the encoder.

Figure 1 shows the main procedures for WSQ encoding and decoding. The same tables specified for an encoder to use to compress a particular image must be provided to a decoder to reconstruct that image.

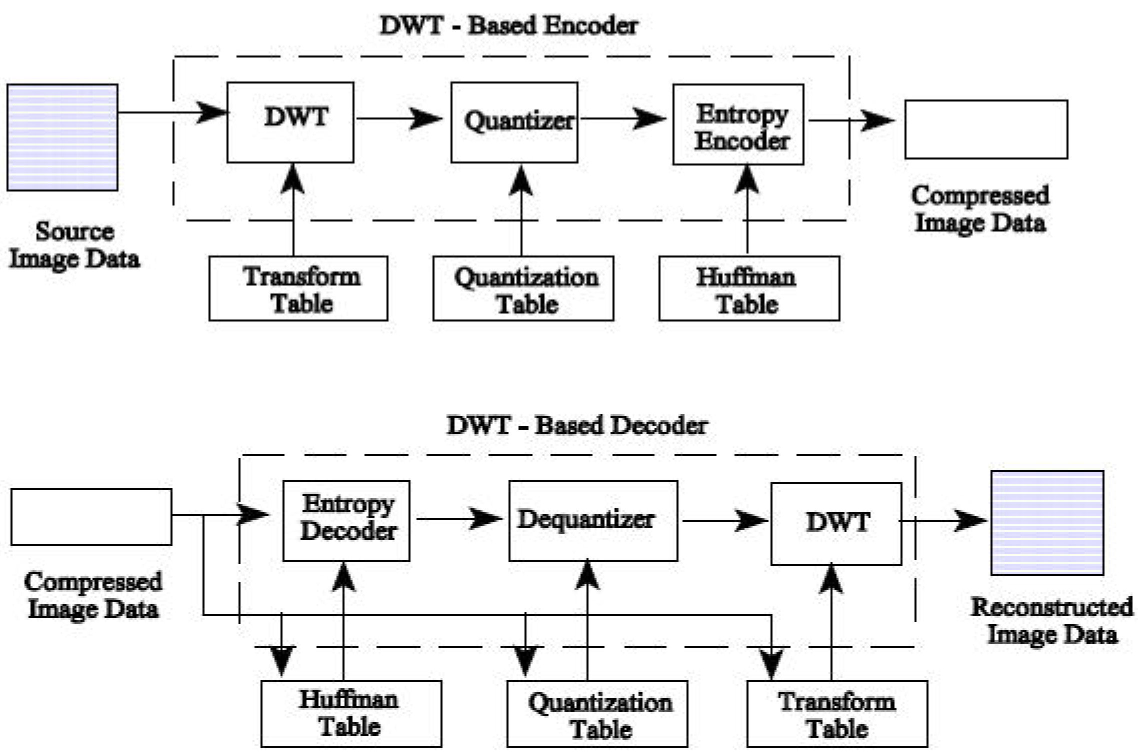


Figure 1. DWT-based encoder and decoder simplified diagram

2.2 Structure of compressed data

Compressed image data is described by a uniform structure and a set of parameters. The various parts of the compressed image data are identified by special two-byte codes called *markers*. Some markers are followed by particular sequences of parameters such as table specifications and headers. Others are used without parameters for functions such as marking the start-of-image and end-of-image. When a marker is associated with a particular sequence of parameters, the marker and its parameters comprise a *marker segment*.

The data created by the *entropy encoder* are also segmented, and one particular marker - the *restart marker* - is used to isolate *entropy-coded data segments*. The encoder outputs the restart markers, intermixed with the entropy-coded data, between certain subband boundaries. Restart markers can be identified without having to decode the compressed data to find them. Because they can be independently decoded, entropy-coded data segments provide for progressive transmission, and isolation of data corruption.

2.3 Interchange format

In addition to certain required marker segments and the entropy-coded segments, the interchange format shall include the marker segments for all filter coefficient, quantization, and entropy-coding tables needed by the decoding process. This guarantees that a compressed image can cross the boundary between identification systems, regardless of how each environment internally associates tables with compressed image data.

2.4 Abbreviated format for compressed image data

The abbreviated format for compressed image data is identical to the interchange format, except that it does not include all tables required for decoding (it may include some of them). This format is intended for use within applications where alternative mechanisms are available for supplying some or all of the table-specification data needed for decoding.

2.5 Abbreviated format for table-specification data

This format contains only table-specification data. It is a means by which the application may install in the decoder the tables required to subsequently reconstruct one or more fingerprint images.

3 Definitions

For the purposes of this Specification, the following definitions apply.

bit stream: Partially encoded or decoded sequence of bits comprising an entropy-coded segment.

byte stuffing: A procedure in which the Huffman coder inserts a zero byte into the entropy-coded segment following the generation of an encoded hexadecimal X'FF' byte.

coding model: A procedure used to convert input data into symbols to be coded.

coding process: A general term for referring to an encoding process, a decoding process, or both.

columns: Samples per line in an image.

compressed data: Either compressed image data or table specification data or both.

compressed image data: A coded representation of an image, as specified in this Specification.

compression: Reduction in the number of bits used to represent source image data.

continuous-tone image: An image sampled at 8 bits per pixel.

decoder: An embodiment of a decoding process.

decoding process: A process which takes as its input compressed image data and outputs a continuous-tone image.

dequantization: The inverse procedure to quantization by which the decoder recovers a representation of the DWT coefficients.

(digital) reconstructed image (data): A continuous-tone image which is the output of the decoder defined in this Specification.

(digital) source image (data): A continuous-tone image used as input to any encoder defined in this Specification.

(digital) image: A two-dimensional array of data.

downsampling: A procedure by which the spatial resolution of an image is reduced.

DWT: (discrete wavelet transform) A linear transformation, implemented by a multirate filter bank, that maps a digital input signal to a collection of output subbands.

encoder: An embodiment of an encoding process.

encoding process: A process which takes as its input a continuous-tone image and outputs compressed image data.

entropy-coded (data) segment: An independently decodable sequence of entropy encoded bytes of compressed image data.

entropy decoder: An embodiment of an entropy decoding procedure.

entropy decoding: A lossless procedure which recovers the sequence of symbols from the sequence of bits produced by the entropy coder.

entropy encoder: An embodiment of an entropy encoding procedure.

entropy encoding: A lossless procedure which converts a sequence of input symbols into a sequence of bits such that the average number of bits per symbol approaches the entropy of the input symbols.

Huffman decoder: An embodiment of a Huffman decoding procedure.

Huffman decoding: An entropy decoding procedure which recovers the symbol from each variable length code produced by the Huffman encoder.

Huffman encoder: An embodiment of a Huffman encoding procedure.

Huffman encoding: An entropy encoding procedure which assigns a variable length code to each input symbol.

Huffman table: The set of variable length codes required in a Huffman encoder and Huffman decoder.

image data: Either source image data or reconstructed image data.

interchange format: The representation of compressed image data for exchange between application environments.

lossless: A descriptive term for encoding and decoding processes and procedures in which the output of the decoding procedure(s) is identical to the input to the encoding procedure(s).

marker: A two-byte code in which the first byte is hexadecimal X'FF' and the second byte is a value between 1 and hexadecimal X'FE'.

marker segment: A marker and associated set of parameters.

parameters: Fixed length integers 8, 16, or 32 bits in length, used in the compressed data format.

procedure: A set of steps which accomplishes one of the tasks which comprise an encoding or decoding process.

progressive (coding): The separation of data segments into blocks that can be transmitted successively to allow the compressed image data to be decoded at successively higher levels of resolution.

quantization table: The set of quantization values (i.e., bin widths) used to quantize DWT coefficients within the subbands.

quantize: The act of performing the quantization procedure for a DWT coefficient.

reference inverse SWT: A double precision floating point implementation of the inverse SWT defined in Annex A of this Specification.

reference SWT: A double precision floating point implementation of the SWT defined in Annex A of this Specification.

restart interval: The number of coefficients processed as an independent sequence within an image.

restart marker: The marker that separates two restart intervals in an image.

run (length): Number of consecutive symbols of the same value.

SWT: (symmetric wavelet transform) A linear transform implemented by applying a DWT to a periodized symmetric extension of the input signal.

sample: One element in the two-dimensional array which comprises a fingerprint image.

table specification data: The coded representation from which the tables, used in the encoder and decoder, are generated.

upsampling: A procedure by which the spatial resolution of an image is increased.

4 Requirements

4.1 Interchange format requirements

The interchange format is the coded representation of compressed image data for exchange between application environments.

The interchange format requirements are that any compressed image data represented in interchange format shall comply with the syntax and codes assignments for the decoding process, as specified in Annex B.

4.2 Encoder requirements

An encoder process converts source fingerprint images to compressed image data. An encoder is an embodiment of the encoding process specified in Annex A. To comply with this Specification, an encoder shall satisfy at least one of the following two requirements:

1) convert source fingerprint image data to compressed image data which complies with the interchange format syntax specified in Annex B with proper accuracy;

2) convert source fingerprint image data to compressed image data which complies with the abbreviated format syntax for compressed image data specified in Annex B with proper accuracy.

The compliance tests for the above requirements are specified in Part 2 of this Specification.

NOTE - There is **no requirement** in this Specification that any encoder which embodies the encoding process specified in Annex A shall be able to operate for all ranges of the parameters which are allowed. An encoder is only required to meet the compliance tests specified in Part 2, and to generate the compressed data format according to Annex B for those parameter values which it does use.

4.3 Decoder requirements

A decoding process converts compressed image data to reconstructed image data. A decoder is an embodiment of the decoding process specified in Annex A. To comply with this Specification, a decoder shall satisfy all three of the following requirements:

1) convert to reconstructed fingerprint image data any compressed image data with parameters that comply with the interchange format syntax specified in Annex B with proper accuracy;

2) accept and properly store any table-specification data which complies with the abbreviated format syntax for table-specification data specified in Annex B; and,

3) convert to reconstructed fingerprint data any compressed image data which complies with the abbreviated format syntax for compressed image data specified in Annex B with proper accuracy, provided that the table-specification data required for decoding the compressed image data has previously been installed into the decoder.

The compliance tests for the above requirements are specified in Part 2 of this Specification.

Annex A. Mathematical definitions

A.1. Source fingerprint image

Source fingerprint images shall be captured with 8 bits of precision per pixel. Before the encoding process computes the discrete wavelet transform (DWT) for the image, the samples, $I(m,n)$, shall be transformed in accordance with the following equation:

$$I'(m,n) = \frac{[I(m,n) - M]}{R} \qquad \begin{matrix} 0 \le m \le Y - 1 \\ 0 \le n \le X - 1 \end{matrix}$$

The image width (X) and height (Y) parameters are defined in Annex B.2.2. The decoding process shall apply an inverse transformation to restore the samples to their original scale. The midpoint and rescale parameters, M and R, are specified by the encoder and transmitted in the compressed image data.

A.2. Subband Coding of Fingerprint Images

A.2.1. Two-Channel Subband Coder (in one dimension)

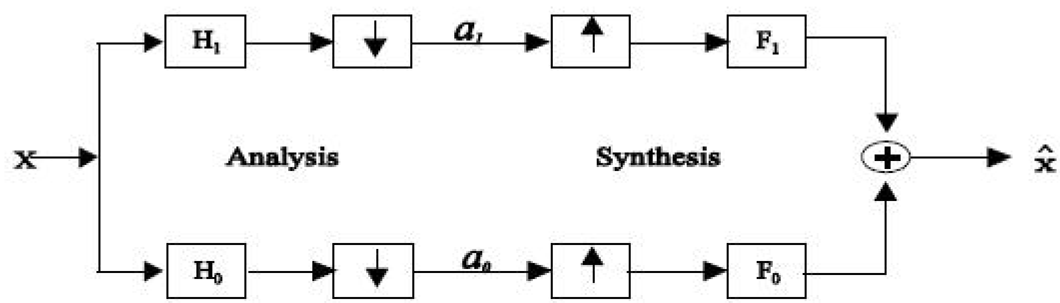


Figure A.1. Two-Channel Subband Coder

A two-channel subband encoder is a digital filter bank of the type shown in Figure A.1. It should be regarded as a pair of systems, the analysis bank and the synthesis bank. The subband coder provides zero distortion, $\hat{x}(n) = x(n)$.

The boxes H_i and F_i denote linear time-invariant digital filters [7], while ↓ and ↑ denote 2:1 down- and up-sampling operations:

$$(x * h)(k) = \sum_n x(n)h(k-n),$$

$$(\downarrow y)(k) = y(2k),$$

$$(\uparrow a)(k) = \begin{Bmatrix} a(k/2), & k \ even \\ 0, & k \ odd \end{Bmatrix}$$

The transform defined by the analysis bank, $x \rightarrow \{a_0 , a_1\}$, will be referred to as a one-dimensional DWT, and the transform given by the synthesis bank as the inverse DWT.

A.2.1.1. Linear Phase Wavelet Filters

This standard utilizes two distinct classes of linear phase finite impulse response (FIR) filters. We let h_0 denote the lowpass filter and h_1 the highpass filter in a filter bank. The first class will contain pairs of odd-length, symmetric filters (i.e., filters whose impulse responses are symmetric about their middle sample). These are called "Type I" linear phase FIR filters, [7], or whole-sample symmetric (WSS) filters. The second class will contain pairs of even-length filters, one symmetric (the lowpass filter) and one antisymmetric (the highpass filter). These are called, respectively, "Type II" and "Type IV" filters, [7]; since such filters are symmetric about the point halfway between their middle two samples, we shall also refer to them as half-sample symmetric/antisymmetric (HSS/HSA or HS-type) filters.

The compressed data format described in Annex B provides only for the transmission of impulse response coefficients from the right halves of the analysis filters; the synthesis filters are completely determined by the following anti-aliasing relations:

$$f_0(n) = (-1)^n h_1(n-1) \quad and$$

$$f_1(n) = (-1)^{n-1} h_0(n-1),$$

For a WSS analysis bank, the lowpass filter, h_0, shall be symmetric about 0, i.e., h_0 runs from $h_0(-r_0)$ to $h_0(r_0)$. Using the syntax of Annex B.2.4.1, the length of h_0 is L0 = 2 r_0+l. The transform table specified in B.2.4.1 contains the impulse response coefficients from the right half of h_0:

$$H0_1 = h_0(0), H0_2 = h_0(1), \ldots, H0_{last} = h_0(t_0).$$

The left half of h_0 is given by the symmetry relation $h_0(-n) = h_0(n)$. The highpass filter, h_1, in a WSS analysis bank shall be symmetric about -1. The transmitted coefficients are:

$$H1_1 = h_1(-1), H1_2 = h_1(0), \ldots, H1_{last} = h_1(r_1 - 1),$$

where L1 = 2r_l+1; the left half of h_1 is given by the symmetry relation $h_1(-1-n) = h_1(n-1)$.

For an HS-type analysis bank, both filters shall be centered at -1/2, and thus run from $h_i(-r_i)$ to $h_i(r_i-1)$, where the length of h_i is Li = 2r_i. The transmitted values are: (i = 0,1)

$$Hi_1 = h_i(0), Hi_2 = h_i(1), \ldots, Hi_{last} = h_i(r_i - 1).$$

The lowpass filter, h_0, is symmetric (HSS), so the left half of h_0 is given by the symmetry relation $h_0(-1-n) = h_0(n)$. The highpass filter, h_1, is antisymmetric (HSA), so the left half of h_1 is given by the symmetry relation $h_1(-1-n) = -h_1(n)$.

A.2.1.2. Constraints on Filter Length

Encoders and decoders shall be capable of forming (or inverting) the subband decomposition specified in Figure A.5 using filters of lengths up to and including the maximum values

L_{max} = 31 for WS-type filters,
L_{max} = 32 for HS-type filters.

Maximum and minimum image dimensions for compliance testing are specified in Part 2.

A.2.2. Symmetric Boundary Conditions for the DWT

The generic input, $x(n)$, to Figure A.1 will, in practice, be a row or column vector from an image or from one of its DWT subbands. To describe precisely how a finite-length signal is transformed by the system depicted in Figure A.1, we use the following conventions for indexing and extrapolating x. All WSQ decoders shall be capable of decoding a compressed signal encoded in accordance with these conventions. x is assumed to run from $x(0)$ to $x(N_0-1)$, where N_0 is the (generic) length of x.

For transformation by WSS filters, x is extended to a whole-sample symmetric signal, $y = E_s^{(1,1)}x$, of length $N = 2N_0 - 2$, and periodized. For HS-type filters, x is extended to a half-sample symmetric signal, y

$= E_s^{(2,2)}x$, of length $N = 2N_0$, and periodized. In each case, the filters are extended with zeros to length N and applied by N-periodic circular convolution; see Figure A.2.

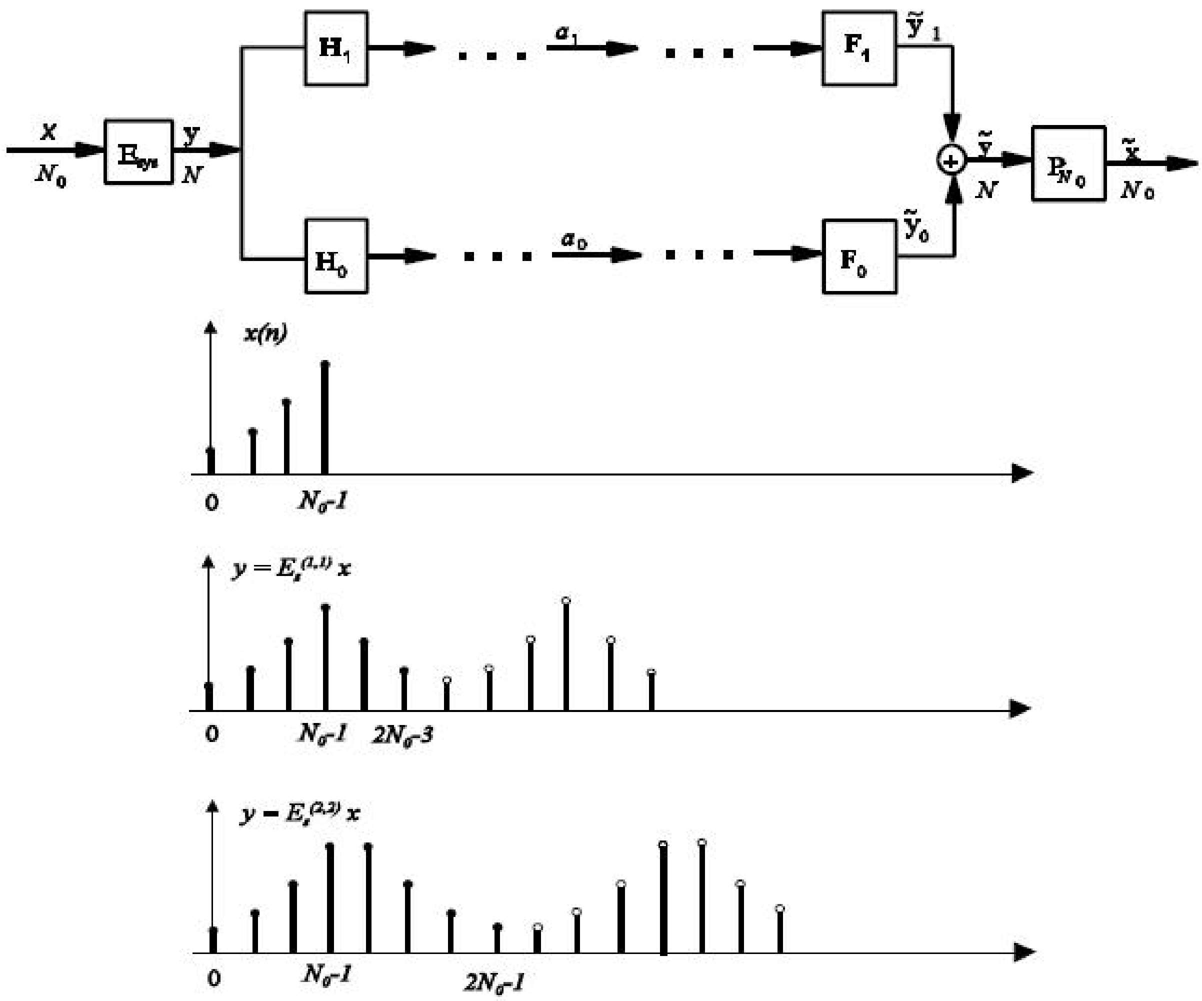


Figure A.2. Symmetric DWT Extension

With the choices of filter and signal symmetries described above, this system generates symmetric DWT subbands, of which only the first half (now denoted a_i) needs to be computed and stored. For instance, with WSS filters and N_0 even, the coefficients

$$a_i(k) = \sum_{n=0}^{N-1} y(n) h_i(2k - n)$$

only need to be computed and stored for $k = 0 ,..., N_0/2 - 1$, even though

$$b_i = \downarrow (y * h_i)$$

has period $N/2 = N_0$-1. This is possible because b_i is itself symmetric and can be reconstructed from the first $N_0/2$ values, $a_i\,(k)$.

The sequences ai generated in this fashion can in turn be extended to whole- or half-sample symmetric signals and cascaded back through the analysis bank, Figure A.2, to achieve a multiband decomposition of the input. The composition of mappings

$$\boldsymbol{x} \xrightarrow{E_{sys}} \boldsymbol{y} \xrightarrow{DWT} \{\boldsymbol{a_0}, \boldsymbol{a_1}\}$$

will be referred to as a one-dimensional symmetric wavelet transform (SWT) [3]. Note that, in spite of the extension of the input signal, x, the SWT still maps an input of length N_0 to a pair of subbands $\{a_0, a_1\}$ containing a total of just N_0 values. When N_0 is odd, a_0 will have length $(N_0 + 1)/2$, and a_1 will have length $(N_0 - 1)/2$ for both WSS and HS-type extensions.

A.2.2.1. Symmetric Subband Synthesis

This section describes the symmetry properties of the subbands $b_i = \downarrow(y*h_i)$, giving the number of nonredundant samples that need to be transmitted,

$$a_i(k) = b_i(k); \quad 0 \leq k \leq \rho_i - 1,$$

and specifies the procedures for extending the quantized transmitted coefficients in the decoding process:

$$\hat{b}_i = E_i \hat{a}_i; \quad \hat{y}_i = f_i * (\uparrow \hat{b}_i).$$

We begin by introducing some terminology for describing symmetric signals like those shown in Figure A.2. If an N-periodic signal is symmetric about n=0 then it is also necessarily symmetric about $n = N/2$; cf. $y = E_s^{(1,1)}x$ in Figure A.2 for $N = 2N_0$-2. When N is even, such signals are called "(1,1)-symmetric" since they are whole-sample symmetric about both centers. Similarly, an N-periodic signal symmetric about -1/2 is also necessarily symmetric about $(N$-1)/2; cf. $y = E_s^{(2,2)}x$ in Figure A.2 for N=2N_0. When N is even, such signals are called "(2,2)-symmetric" since they are half-sample symmetric about both centers. When N is odd, we get signals that are WSS about one center and HSS about the other; such signals are called "(1,2)-symmetric" if they are WSS about 0, and "(2,1)-symmetric" if they are HSS about -1/2. There are obvious antisymmetric analogues of these symmetry properties.

Given a signal $w(n)$, $0 \leq n \leq K$-1, let $E_s^{(i,j)}w$ (resp., $E_a^{(i,j)}w$) denote the (i,j)-symmetric (resp., (i,j)-antisymmetric) extension of $w(n)$, where i,j = 1 or 2, generalizing the two extensions shown in Figure A.2 for w=x. If a subband $b(k)$ is (i,j)-symmetric and

$$a(k) = b(k); \quad 0 \leq k \leq \rho - 1$$

is a complete, nonredundant half-period of b, then b can be reconstructed from a via the extension $b = E_s^{(i,j)}a$. A similar statement holds for antisymmetric subbands. Since the symmetryof b is completely determined by the symmetry of the extension $y = E_{sys}x$ and the symmetry of the analysis filter, h, it suffices to quantize and transmit only the half-period, a, reconstructing b in the decoder using a known extension operator, E. This method of applying a DWT filter bank to a finite-duration input signal, x, is referred to as the symmetric wavelet transform (SWT) algorithm; a detailed treatment is presented in [3].

Table A.1 lists the symmetry properties of the subbands, b, and their "ranks"ρ, which specifies the number of coefficients, $a(k)$, that need to be transmitted. The table is divided into two cases: one case for WSS filter banks, which use the analysis extension $y = E_s^{(1,1)}x$ (the "(1,1)-SWT"), and a second case for HS-type filter banks, which use the analysis extension $y = E_s^{(2,2)}x$ (the "(2,2)-SWT"). These filter banks are described above in A.2.1.1.

		Case 1: WSS Filters		Case 2: HS-Type Filters	
		Input Length, N_0		Input Length, N_0	
		Even	Odd	Even	Odd
Filter	h_0	(1,2) - sym. $\rho_0 = N_0/2$	(1,1) - sym. $\rho_0 = (N_0+1)/2$	(2,2) - sym. $\rho_0 = N_0/2$	(2,1) - sym. $\rho_0 = (N_0+1)/2$
	h_1	(2,1) - sym. $\rho_1 = N_0/2$	(2,2) - sym. $\rho_1 = (N_0-1)/2$	(2,2) - antisym. $\rho_1 = N_0/2$	(2,1) - antisym. $\rho_1 = (N_0-1)/2$

Table A.1. Symmetry, Rank of SWT Subbands

A.2.3. Wavelet Decomposition in Two Dimensions

The tree structure for a single level of a 2-dimensional image decomposition system is depicted in Figure A.3. The row vectors of the image are filtered by applying the SWT algorithm described in the preceding section. The same procedure is then applied to the column vectors of the resulting array, giving a decomposition into four subbands, as shown in Figure A.4. Note that the pair of indices indicates which filters were applied to the rows and columns of the signal. For instance, a_{10} has been highpass-filtered on rows and lowpass-filtered on columns, so it contains vertical edge features. The filters applied in succeeding levels of filter bank cascade are indicated with succeeding pairs of binary indices.

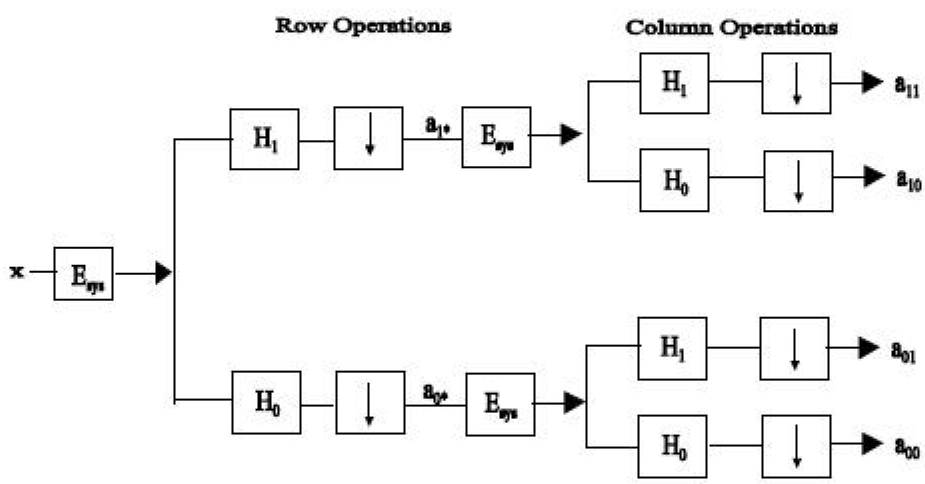


Figure A.3. Single-Level, Two-Dimensional Subband Analysis

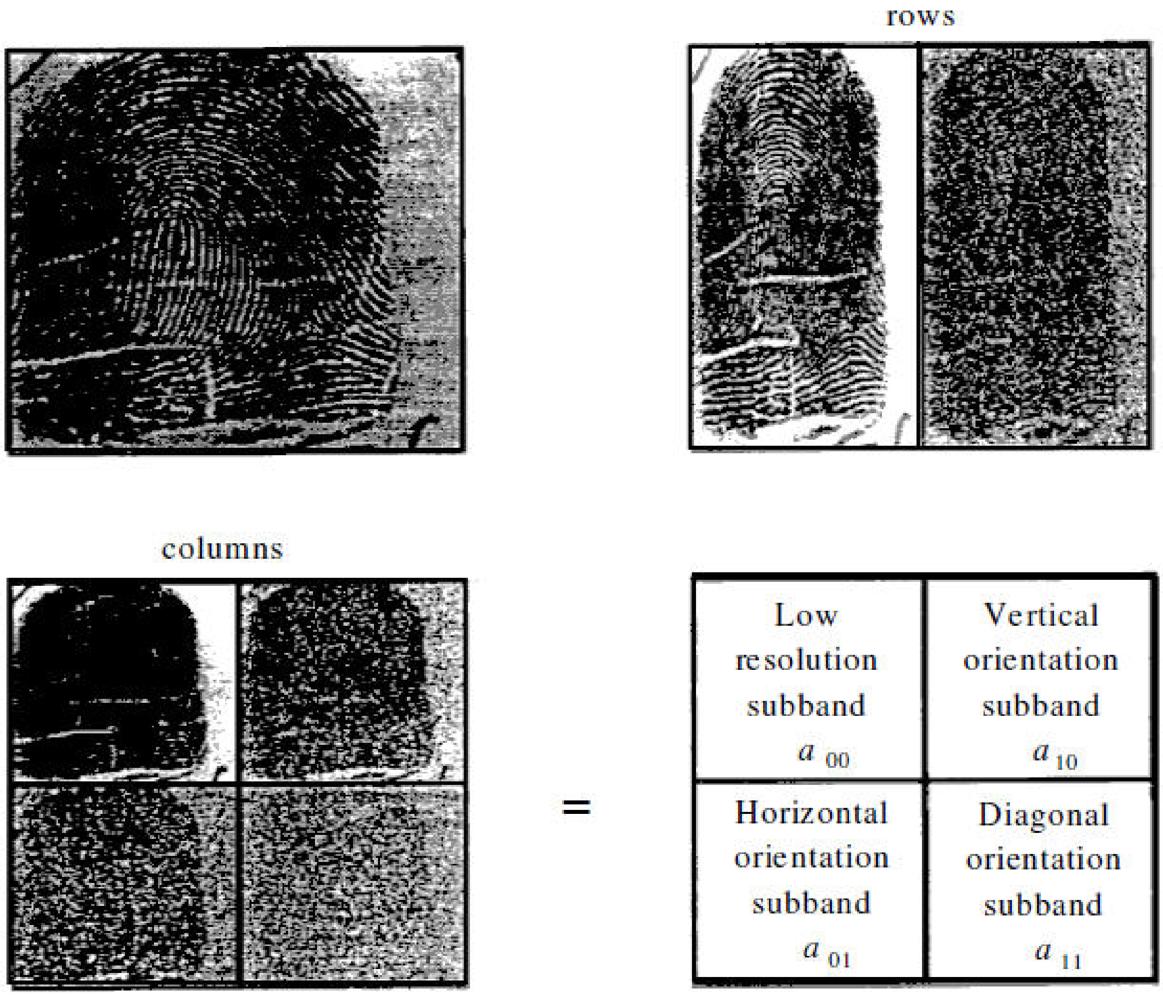
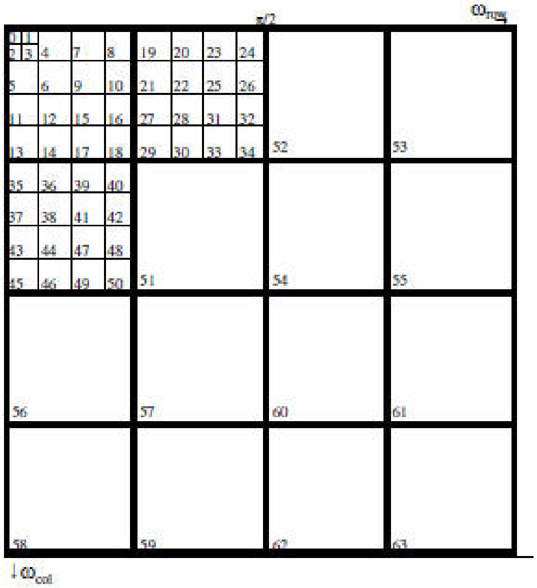


Figure A.4. Four Subband Decomposition

A.2.4. Subband Structure

After the first level of decomposition shown in Figure A.3, any of the resulting four subbands may be cascaded back through the filter bank to further split the subband into four more subbands. This process is continued until the desired subband structure is obtained. Figure A.5 is the subband structure specified for fingerprint images by this Specification. The table shows which filters are applied to obtain each subband.



Filter Bank Path

0	00,00,00,00,00	22	00,10,10,11	44	00,01,00,11
1	00,00,00,00,10	23	00,10,00,10	45	00,01,00,00
2	00,00,00,00,01	24	00,10,00,00	46	00,01,00,10
3	00,00,00,00,11	25	00,10,00,11	47	00,01,10,11
4	00,00,00,10	26	00,10,00,01	48	00,01,10,01
5	00,00,00,01	27	00,10,11,01	49	00,01,10,10
6	00,00,00,11	28	00,10,11,11	50	00,01,10,00
7	00,00,10,10	29	00,10,11,00	51	00,11
8	00,00,10,00	30	00,10,11,10	52	10,10
9	00,00,10,11	31	00,10,01,11	53	10,00
10	00,00,10,01	32	00,10,01,01	54	10,11
11	00,00,01,01	33	00,10,01,10	55	10,01
12	00,00,01,11	34	00,10,01,00	56	01,01
13	00,00,01,00	35	00,01,01,00	57	01,11
14	00,00,01,10	36	00,01,01,10	58	01,00
15	00,00,11,11	37	00,01,01,01	59	01,10
16	00,00,11,01	38	00,01,01,11	60	11,11
17	00,00,11,10	39	00,01,11,10	61	11,01
18	00,00,11,00	40	00,01,11,00	62	11,10
19	00,10,10,00	41	00,01,11,11	63	11,00
20	00,10,10,10	42	00,01,11,01		
21	00,10,10,01	43	00,01,00,01		

Figure A.5. Subband Decomposition

A.3. Quantization

After the subband decomposition is computed the resulting coefficients are quantized uniformly within subbands. The quantizer zero bin width and step size for each subband are contained in the quantization table. A quantization step size of zero ($Q_k = 0$) indicates that all coefficients within the subband are zero and the subband is not transmitted. The following equation would apply to the wavelet coefficients $a_k(m,n)$ in subband k.

$$p_k(m,n) = \begin{cases} \left\lfloor \dfrac{a_k(m,n) - Z_k/2}{Q_k} \right\rfloor + 1, & a_k(m,n) > Z_k/2 \\ 0, & -Z_k/2 \le a_k(m,n) \le Z_k/2 \\ \left\lceil \dfrac{a_k(m,n) + Z_k/2}{Q_k} \right\rceil - 1, & a_k(m,n) < -Z_k/2 \end{cases}$$

At the decoder**,** the following equation dequantizes the indices. The value C determines the quantization bin centers.

$$\hat{a}_k(m,n) = \left\{ \begin{array}{ll} (p_k(m,n) - C)Q_k + Z_k/2, & p_k(m,n) > 0 \\ 0, & p_k(m,n) = 0 \\ (p_k(m,n) + C)Q_k - Z_k/2, & p_k(m,n) < 0 \end{array} \right\}$$

Z_k is the width of the center (zero) quantization bin and Q_k is the width of the nonzero quantization bins in the k^{th} subband. The function $\lceil x \rceil$ is the least integer greater than or equal to x, and $\lfloor x \rfloor$ is the greatest integer less than or equal to x.

A.4. Entropy Coding

A.4.1. Data Sequence

Consecutively numbered subbands are combined into blocks for entropy coding. To provide for progressive transmission the image shall be divided into a minimum of three blocks, with the first break between subbands 18 and 19, and the second between subbands 51 and 52. Additional subdivision of these three blocks is optional. All the subbands within a block must use the same Huffman encoding table. Within a subband the coefficients shall be ordered left to right, top to bottom. Within a block, subbands shall be listed consecutively in increasing order.

A.4.2. Huffman Encoder

A Huffman encoder is used to assign variable length codes to the quantized coefficients within a block. Special codes are provided for zero runs. Coefficients and zero run lengths outside the range of the table are embedded in the code stream with an escape sequence. A maximum of 8 tables may be installed in the decoder. Table A.2 lists the complete symbol set.

position	value
1	zero run length 1
2	zero run length 2
3	zero run length 3
.	
100	zero run length 100
101	esc for pos 8 bit coeff
102	esc for neg 8 bit coeff
103	esc for pos 16 bit coeff
104	esc for neg 16 bit coeff
105	esc for zero run - 8 bits
106	esc for zero run - 16 bits
107	coeff value 73
108	coeff value -72
109	coeff value -71
.	
180	*- use position 1 only -*
.	
253	coeff value 73
254	coeff value 74

Table A.2. Huffman Table Input Symbols

Annex B. Compressed data formats

This Annex Specifies three compressed data formats:

1) the interchange format, specified in B.1 and B.2;

2) the abbreviated format for compressed image data, specified in B.3;

3) the abbreviated format for table-specification data, specified in B.4;

Section B.1 describes the constituent parts of these formats. The format specifications in Sections Figure B.2 - B.4 give the conventions for symbols and figures used in the format specifications.

B.1. General aspects of the compressed data format specifications

Structurally, the compressed data formats consist of an ordered collection of parameters, markers, and entropy-coded data segments. Parameters and markers in turn are often organized into marker segments. Because all of these constituent parts are represented with byte-aligned codes, each compressed data format consists of an ordered sequence of 8-bit bytes. For each byte, a most significant bit (MSB) and a least significant bit (LSB) are defined.

B.1.1. Constituent parts

This section gives a general description of each of the constituent parts of the compressed data format.

B.1.1.1. Parameters

Parameters are integers, with values specific to the encoding process, source image characteristics, and other features selectable by the application. Parameters are assigned either 4-bit, 1-byte, or 2-byte codes. Except for certain optional groups of parameters, parameters encode critical information without which the decoding process cannot properly reconstruct the image.

The code assignment for a parameter shall be an unsigned integer of specified length in bits with the particular value of the parameter.

For parameters which are 2 bytes (16 bits) in length, the most significant byte shall come first in the interchange format. Parameters which are 4 bits in length always come in pairs, and the pair shall always be encoded in a single byte. The first 4-bit parameter of the pair shall occupy the most significant 4 bits of the byte. Within any 16, 8 or 4 bit parameter, the MSB shall come first and LSB shall come last.

B.1.1.2. Markers

Markers serve to identify the various structural parts of the compressed data formats. Most markers start marker segments containing a related group of parameters; some markers stand alone. All markers are assigned two-byte codes: an X'FF' byte followed by a byte which is not equal to 0 or X'FF' (see Table B.1). Any marker may optionally be preceded by any number of fill bytes, which are bytes assigned code X'FF'.

NOTE - Because of this special code-assignment structure, markers make it possible for a decoder to parse the interchange format and locate its various parts without having to decode other segments of image data.

B.1.1.3. Marker assignments

All markers shall be assigned two-byte codes: an X'FF' byte followed by a second byte which is not equal to 0 or X'FF'. The second byte is specified in Table B.1 for each defined marker. An asterisk (*) indicates a marker which stands alone, that is, which is not the start of a marker segment.

X'FFA0'	SOI *	Start of Image
X'FFA1'	EOI *	End of Image
X'FFA2'	SOF	Start of Frame
X'FFA3'	SOB	Start of Block
X'FFA4'	DTT	Define Transform Table
X'FFA5'	DQT	Define Quantization Table
X'FFA6'	DHT	Define Huffman table(s)
X'FFA7'	DRI	Define Restart Interval
X'FFB0' - X'FFB7'	RSTm*	Restart with modulo 8 count "m"
X'FFA8'	COM	Comment

Table B.1. Marker code Assignments

B.1.1.4. Marker segments

A marker segment consists of a marker followed by a sequence of related parameters. The first parameter in a marker segment is the two-byte length parameter. This length parameter encodes the number of bytes in the marker segment, including the length parameter and excluding the two byte marker. The marker segments identified by the SOF and SOB marker codes are referred to as headers: the frame header and the block header respectively.

B.1.1.5. Entropy-coded data segments

An entropy-coded data segment (ECS) contains the output of an entropy-coding procedure. It consists of an integer number of bytes.

NOTES

1. Making entropy-coded segments an integer number of bytes is achieved as follows:
   1-bits are used, if necessary, to pad the end of the compressed data to complete the final byte of a segment.
2. To ensure that a marker does not occur within an entropy-coded segment**,** any X'FF' byte generated by the Huffman encoder is followed by a "stuffed" zero byte.

B.1.2. Syntax

In Sections B.2 and B.3 the interchange format syntax is specified. For the purposes of this Specification, the syntax specification consists of:

- the required ordering of markers, parameters, and entropy-coded segments;
- identification of optional or conditional constituent parts;
- the name, symbol, and definition of each marker and parameter;
- the allowed values of each parameter;
- any restrictions on the above which are specific to the various coding processes.

The ordering of constituent parts and the identification of which are optional or conditional is specified by the syntax figures in Sections B.2 and B.3. Names, symbols, definitions, allowed values, and restrictions are specified immediately below each syntax figure.

B.1.3. Conventions for syntax figures

The syntax figures in Sections B.2 and B.3 are a part of the interchange format specification. The following conventions, illustrated in Figure B.1, apply to these figures:

- parameter/marker indicator: a thin-lined box encloses either a marker or a single parameter;
- segments indicator: a thick-lined box encloses either a marker segment, an entropy-coded data segment, or combinations of these;
- parameter length indicator: the width of a thin-lined box is proportional to the parameter length (4, 8, or 16 bits, shown as E, B, and D respectively in Figure B.1) of the marker or parameter it encloses; the width of thick-lined boxes is not meaningful;
- optional/conditional indicator: square brackets indicate that a marker or marker segment is optionally or conditionally present in the compressed image data;
- ordering: in the interchange format a parameter or marker shown in a figure precedes all of those shown to its right, and follows all of those shown to its left.

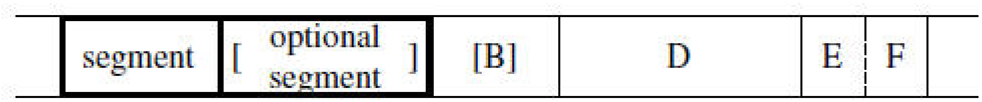


Figure B.1. Syntax notation conventions

B.1.4. Conventions for symbols, code lengths, and values

Following each syntax figure in Sections B.2 and B.3, the symbol, name length, and definition for each marker and parameter shown in the figure is specified.

The following conventions apply to symbols for markers and parameters:

- all marker symbols have three upper-case letters, and some also have a subscript. Examples: SOI, RST_m;
- all parameter symbols have one upper-case letter; some also have one lower-case letter and some have subscripts. Examples: Y, Nf, Q_k.

B.2. General syntax

This section specifies the interchange format syntax which applies to all coding processes specified in this Specification.

B.2.1. High-level syntax

Figure B.2 specifies the order of the high-level constituent parts of the interchange format.

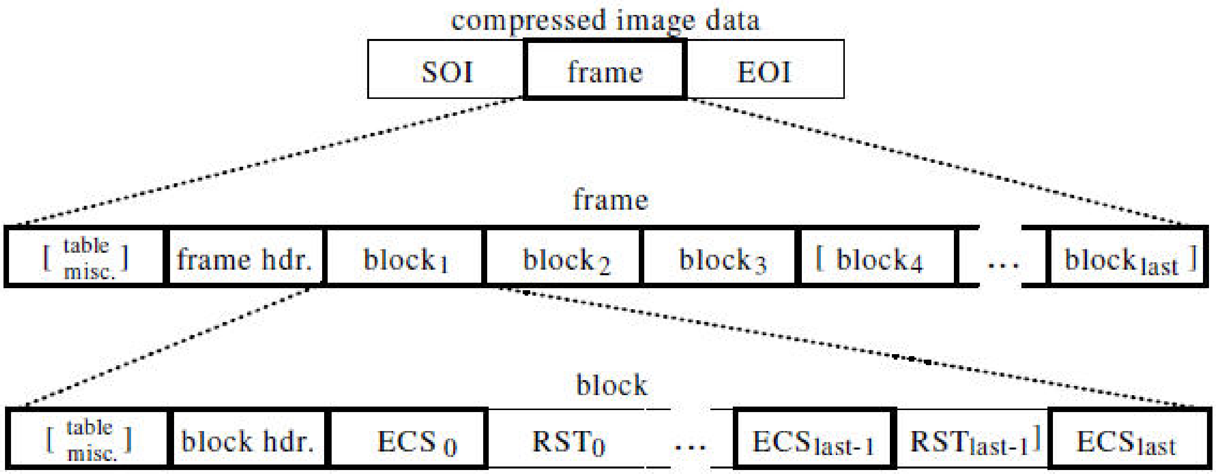


Figure B.2. High-level syntax

The three markers shown in Figure B.2 are defined as follows:

SOI: start of image marker: marks the start of a compressed image represented in the interchange format.

EOI: end of image marker: marks the end of a compressed image represented in the interchange format.

RST_m: restart marker: an optional marker which is placed between entropy-coded segments only if restart is enabled. There are 8 unique restart markers (m=0-7) which repeat in sequence from 0 to 7 to provide a modulo 8 restart interval count.

The top level of Figure B.2 specifies that the interchange format shall begin with an SOI marker, shall contain one frame, and shall end with an EOI marker.

The second level of Figure B.2 specifies that a frame shall begin with a frame header and shall contain one or more blocks. A frame header may be preceded by one or more table-specification or miscellaneous marker segments.

The third level of Figure B.2 specifies that a block shall begin with a block header and shall contain one or more entropy-coded data segments. Each block header may be preceded by one or more table-specification or miscellaneous marker segments. If restart is not enabled, there shall be only one entropy-coded segment (the one labeled "last"), and no restart markers shall be present. If restart is enabled, the number of entropy-coded segments is defined by the size of the image and the defined restart interval. In this case, a restart marker shall follow each entropy-coded segment except the last one.

Figure B.2 specifies the locations where table-specification segments may be present. However, this Specification hereby specifies that the interchange format shall contain all table-specification data necessary for decoding the compressed image. Consequently, the required table-specification data shall be present at one or more of the allowed locations.

B.2.2. Frame header syntax

Figure B.3 specifies the frame header which shall be present at the start of a frame. This header specifies the source image characteristics and encoder version.

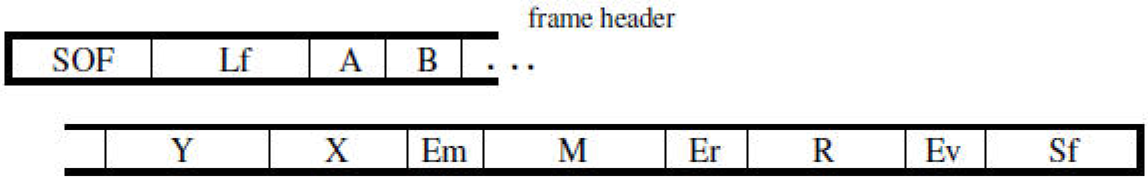


Figure B.3. Frame header syntax

The markers and parameters shown in Figure B.3 are defined below.

SOF: (16 bits) Start of frame marker; marks the beginning of the frame parameters.

Lf: (16 bits) Frame header length; specifies the length of the frame header shown in Figure B.3

A: (8 bits) scanner black calibration value.

B: (8 bits) scanner white calibration value; A and B record the dynamic range calibration of the scanner for use in display systems.

Y: (16 bits) number of lines; specifies the number of lines in the source image.

X: (16 bits) number of samples per line; specifies the number of samples per line in the source image.

Em: (8 bits) scale exponent; the decimal point in M is shifted left Em places.

M: (16 bits) location value for image transformation parameters.

Er: (8 bits) scale exponent; the decimal point in R is shifted left Er places.

R: (16 bits) scale value for image transformation parameters.

Ev: (8 bits) identifies the WSQ encoder algorithm (parameterization) that was used on this image. (binary 2 - 0x02 for FBI Encoder Number Two)

Sf: (16 bits) identifies the software implementation that encoded this image.

B.2.3. Block header syntax

Figure B.4 specifies the block header which shall be present at the start of a block segment. This header specifies the selection of the Huffman coding table used for all the subbands in the block.

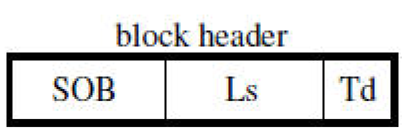


Figure B.4. Block header syntax

The marker and parameters shown in Figure B.4 are defined below.

SOB: (16 bits) start of block marker; marks the beginning of the block header.

Ls: (16 bits) subband header length; specifies the length of the block header shown in Figure B.4.

Td: (8 bits) Huffman coding table selector; selects one of eight possible entropy coding tables needed for decoding the subbands within the segment.

B.2.4. Table-specification and miscellaneous marker segment syntax

Figure B.5 specifies that at the places indicated in Figure B.2, any of the table-specification segments or miscellaneous marker segments specified in B.2.4.1- B.2.4.5 may be present in any order and with no limit on the number of segments.

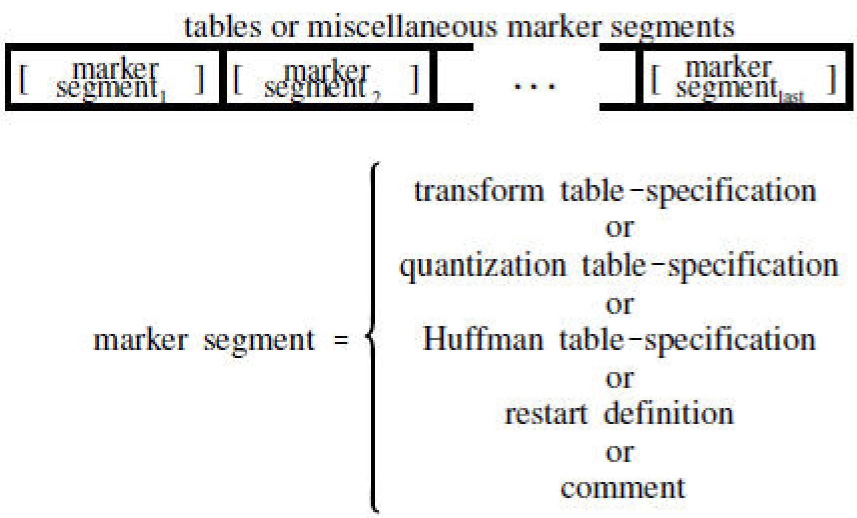


Figure B.5. Table/miscellaneous marker segment syntax

If any table specifications occur in the compressed image data, they shall replace any previous specification, and shall be used whenever the tables are required in the remaining scans in the frame. If a table specification for a given table occurs more than once in the compressed image data, each specification shall replace the previous specification.

B.2.4.1. Transform table-specification

Figure B.6 specifies the marker segment which defines a transform table.

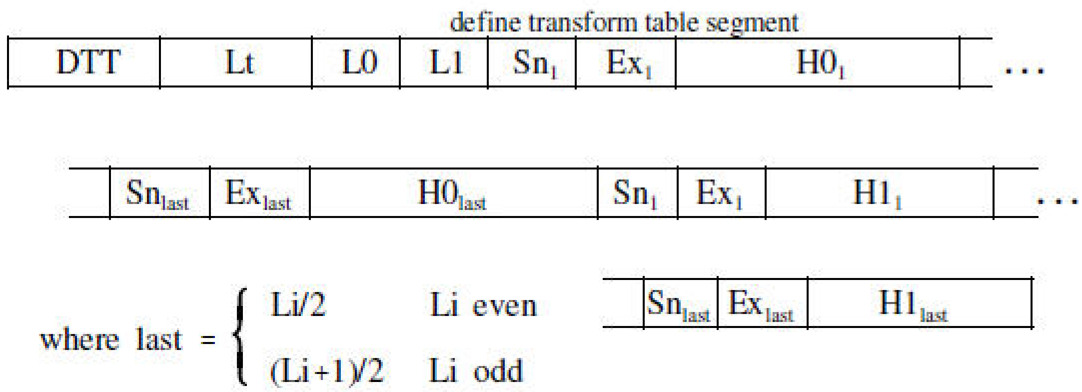


Figure B.6. Transform table syntax

The markers and parameters shown in Figure B.6 are defined below.

DTT: (16 bits) define transform table marker; marks the beginning of transform table-specification parameters.

Lt: (16 bits) transform table definition length; specifies the length of all transform table parameters shown in Figure B.6.

L0: (8 bits) number of analysis lowpass filter coefficients (length of h_o).

L1: (8 bits) number of analysis high pass filter coefficients (length of h_1).

Sn_k: (8 bits) sign of the k^{th} filter coefficient; zero is positive, nonzero is negative.

Ex_k: (8 bits) scale exponent; the decimal point in the k^{th} filter coefficient is moved left Ex places.

$H0_k$: (32 bits) low-pass analysis filter element; specifies the filter low-pass coefficients for the DWT, starting at the center of the filter ($H0_1$).

$H1_k$: (32 bits) High-pass analysis filter element; specifies the filter high-pass coefficients for the DWT, starting at the center of the filter ($H1_1$).

B.2.4.2. Quantization table-specification syntax

Figure B.7 specifies the marker segment which defines a quantization table.

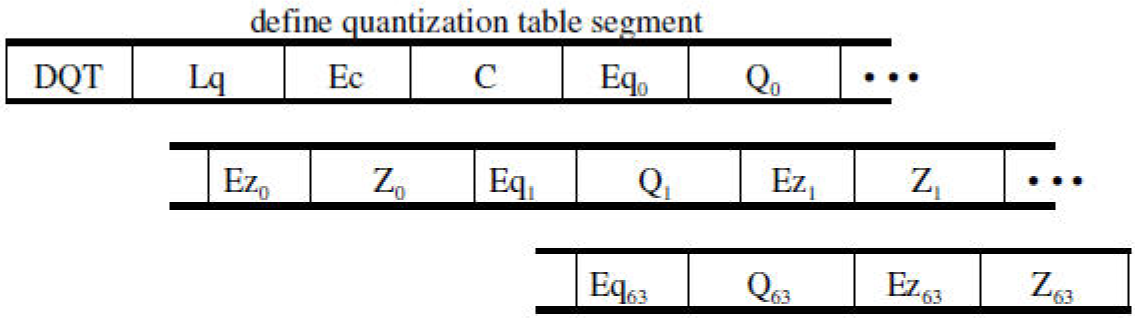


Figure B.7. Quantization table syntax

The markers and parameters shown in Figure B.7 are defined below.

DQT: (16 bits) define quantization table marker; marks the beginning of quantization table-specification parameters.

Lq: (16 bits) quantization table definition length; specifies the length of all quantization table parameters shown in Figure B.7.

Ec: (8 bits) scale exponent; the decimal point in C is moved left Ec places.

C: (16 bits) quantizer bin center parameter.

Eq_k: (8 bits) scale exponent; the decimal point in the k^{th} quantization table element is moved left Eq places.

Q_k: (16 bits) quantization table element; specifies the quantization bin size for the k^{th} subband.

Ez_k: (8 bits) scale exponent; the decimal point in the k^{th} zero bin quantization table element is moved left Ez places.

Z_k: (16 bits) Zero bin table element; specifies the center (zero) bin size for the k^{th} subband.

Quantization table elements shall be specified in subband order, as shown in Figure B.7. Once a quantization table has been defined, it may be used for subsequent images. If a table has never been defined, the results are unpredictable.

B.2.4.3. Huffman table-specification syntax

Figure B.8 specifies the marker segment which defines one or more Huffman table specifications.

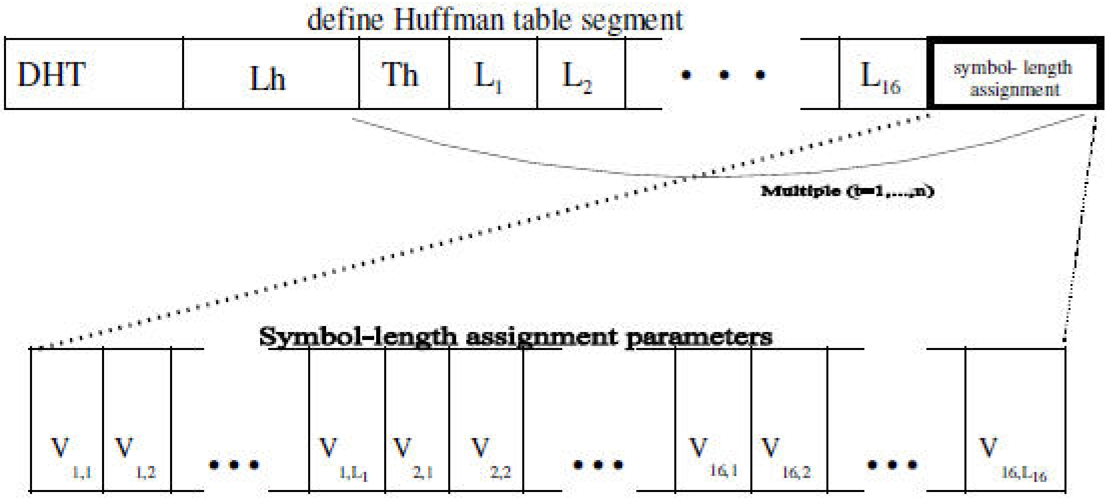


Figure B.8. Huffman table syntax

The markers and parameters shown in Figure B.8 are defined below.

DHT: (16 bits) define Huffman table marker; marks the beginning of Huffman table definition parameters.

Lh: (16 bits) Huffman table definition length; specifies the length of all Huffman table parameters shown in Figure B.8.

Th: (8 bits) Huffman table identifier; specifies one of eight possible destinations at the decoder into which the Huffman table shall be installed.

L_i: (8 bits) number of Huffman codes of length i; specifies the number of Huffman codes for each of the 16 possible lengths allowed by this Specification. The L_i's are the elements of the list BITS.

V_{ij}: (8 bits) value associated with each Huffman code; specifies, for each i, the value associated with each Huffman code of length i. The meaning of each value is determined by the Huffman coding model. The V_{ij}'s are the elements of the list HUFFVAL.

The value n in Figure B.8 is the number of Huffman tables specified in the DHT marker segment. The value m_t is the number of parameters which follow the 16 $L_i(t)$ parameters for Huffman table t, and is given by:

$$m_t = \sum_{i=1}^{16} L_i(t)$$

In general, m_t is different for each table.

Once a Huffman table has been defined, it may be used for subsequent images. If a table has never been defined, the results are unpredictable.

B.2.4.4. Restart interval definition syntax

Figure B.9 specifies the marker segment which defines the restart interval.

Figure B.9. Restart interval segment

The markers and parameters shown in Figure B.9 are defined below.

DRI: (16 bits) define restart interval marker; marks the beginning of the parameters which define the restart interval.

Lr: (16 bits) define restart interval segment length; specifies the length of the parameters in the DRI segment.

Ri: (16 bits) restart interval; specifies the number of coefficients in the restart interval. The SOI marker disables the restart intervals. A DRI marker segment with Ri nonzero shall be present to enable restart interval processing for the following subbands. A DRI marker segment with Ri equal to zero shall disable restart intervals for the following subbands.

B.2.4.5. Comment syntax

Figure B.10 specifies the marker segment structure for a comment segment.

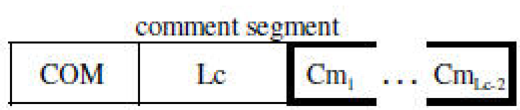


Figure B.10. Comment segment syntax

The markers and parameters shown in Figure B.10 are defined below.

COM: (16 bits) comment marker; marks the beginning of a comment.

Lc: (16 bits) comment segment length; specifies the length of the comment segment shown in Figure B.10.

Cm_i: (8 bits) comment byte; the interpretation is left to the application.

B.3. Abbreviated format for compressed image data

Figure B.2 shows the high-level constituent parts of the interchange format. This format includes all table specifications required for decoding. If an application environment provides methods for table specification other than by means of the compressed image data, some or all of the table specifications may be omitted. Compressed image data which is missing any table specification data required for decoding has the abbreviated format.

B.4. Abbreviated format for table-specification data

Figure B.2 shows the high-level constituent parts of the interchange format. If no blocks are present in the compressed image data, the only purpose of the compressed image data is to convey table specifications or miscellaneous marker segments defined in B.2.4.1, B.2.4.2, B.2.4.3 and B.2.4.5. In this case the compressed image data has the abbreviated format for table specification data as shown in Figure B.11.

Figure B.11. Abbreviated format for table-specification data syntax

Annex C. Huffman table specification

Huffman tables are specified in the interchange format in terms of a 16-byte list (BITS) giving the number of codes for each code length from 1 to 16. This is followed by a list of the 8-bit symbol values (HUFFVAL), each of which is assigned a Huffman code. The symbol values are placed in the list in order of increasing code length. Code lengths greater than 16 bits are not allowed. In addition, the codes shall be generated such that the all-1-bit code word of any length is reserved as a prefix for longer code words.

> NOTE: The order of the symbol values within HUFFVAL is determined only by code length. Within a given code length the ordering of the symbol values is arbitrary.

This Annex specifies the procedure by which the Huffman table (of Huffman code words and their corresponding 8-bit symbol values) are derived from the two lists (BITS and HUFFVAL) in the interchange format. However, the way in which these lists are generated is not specified. The lists should be generated in a manner which is consistent with the rules for Huffman coding, and shall observe the constraints discussed in the previous paragraph.

> NOTE: There is no requirement in this Specification that any encoder or decoder shall implement the procedures in precisely the manner specified by the flow charts in this Annex. The sole criterion for an encoder or decoder to be considered in compliance with this Specification is that it satisfy the requirements given in Part 1, as determined by the compliance tests specified in Part 2.

C.1. Marker segments for Huffman table specification

The DHT marker identifies the start of Huffman table definitions within the compressed image data. B.2.4.3 specifies the syntax for Huffman table specification.

C.2. Conversion of Huffman tables specified in interchange format to tables of codes and lengths

Given a list BITS (1..16) containing the number of codes of each size, and a list HUFFVAL containing the symbol values to be associated with those codes as described above, two tables are generated. The HUFFSIZE table contains a list of code lengths; the HUFFCODE table contains the Huffman codes corresponding to those lengths.

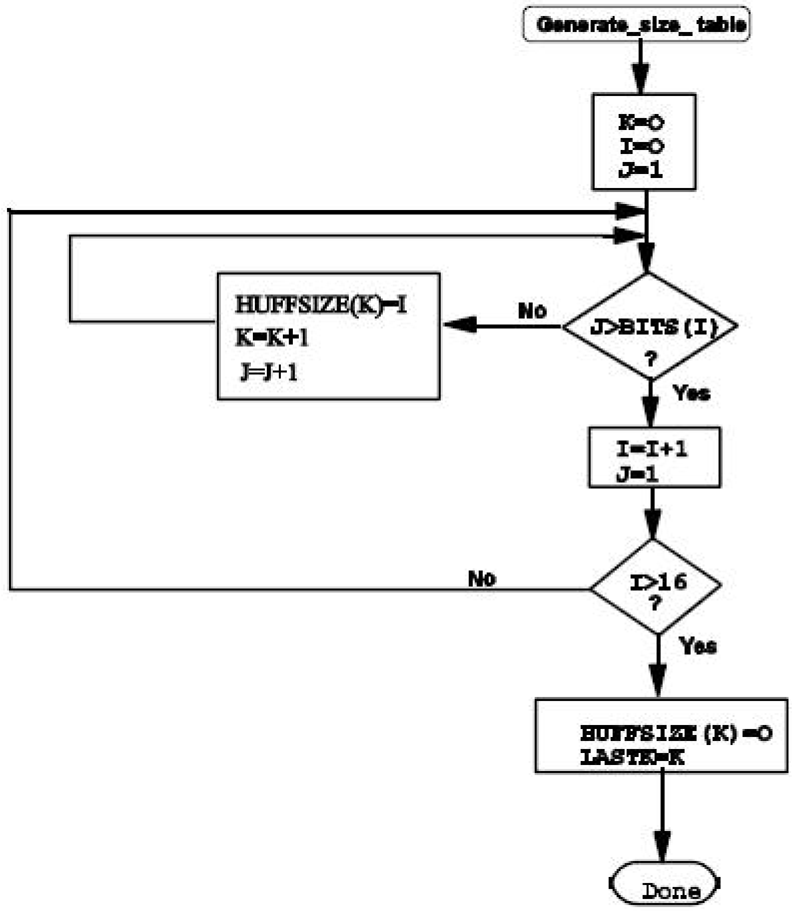


Figure C.1. Generation of table of Huffman code sizes

Note that the variable LASTK is set to the index of the last entry in the table. A Huffman code table, HUFFCODE, containing a code for each size in HUFFSIZE is generated by the procedure in Figure C.2.

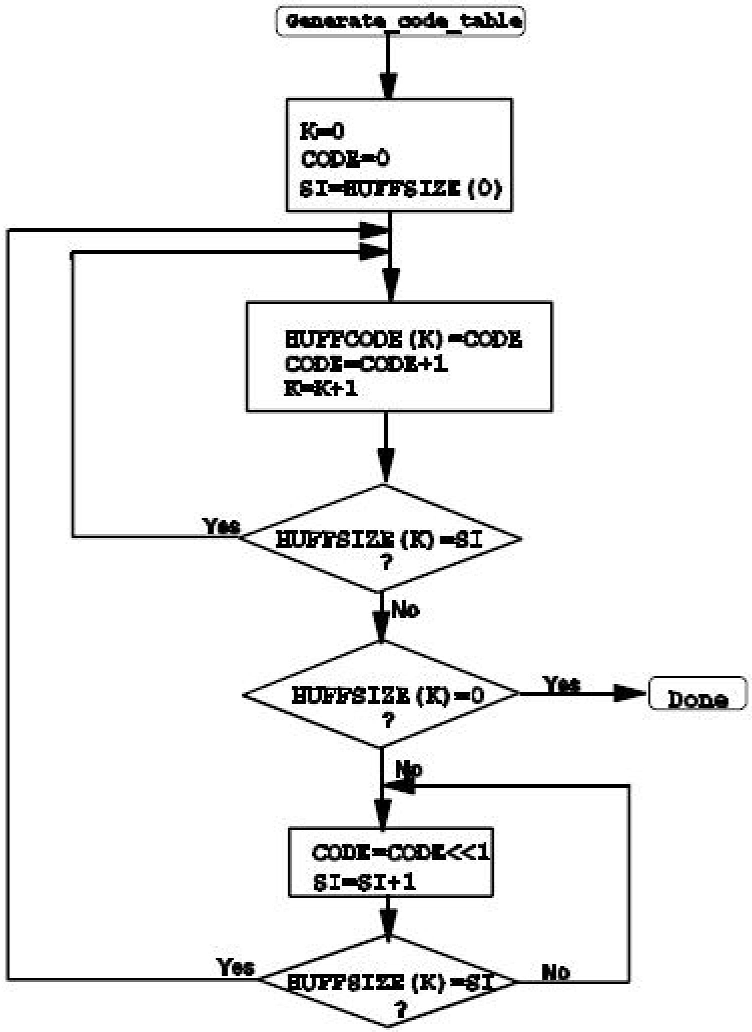


Figure C.2. Generation of table of Huffman codes

Two tables, HUFFCODE, and HUFFSIZE, have now been initialized. The entries in the tables are ordered according to increasing Huffman code numeric value and length.

The encoding procedure code tables, EHUFCO and EHUFSI, are created by reordering the codes specified by HUFFCODE and HUFFSIZE according to the symbol values assigned to each code in HUFFVAL. Figure C.3 illustrates this ordering procedure.

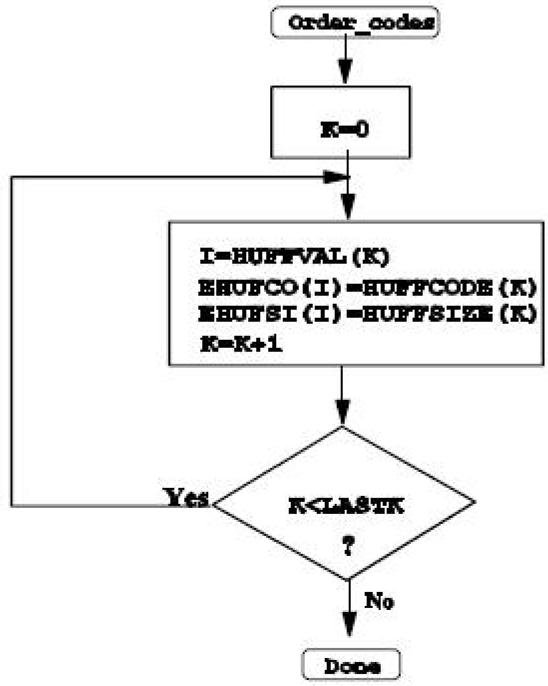


Figure C.3. Ordering procedure for encoding procedure code tables

C.3. Bit ordering within bytes

The root of a Huffman code is placed toward the MSB of the byte, and successive bits are placed in the direction MSB to LSB of the byte. Remaining bits, if any, go into the next byte following the same rules.

Annex D. Bibliography

1. M. Antonini, M. Barlaud, P. Mathieu and I. Daubechies, "Image Coding Using Wavelet Transform", *IEEE Trans. Image Processing*, vol. 1, pp. 205-220, April 1992.
2. J. N. Bradley, C. M. Brislawn, "1992 Progress Report: Compression of Fingerprint Data Using the Wavelet Vector Quantization Image Compression Algorithms", Los Alamos National Laboratory, April, 1992, Tech. Report #LA-UR-92-1507.
3. M. Brislawn, "Classification of Symmetric Wavelet Transforms", Los Alamos National Laboratory, Aug. 1992, Tech. Report #LA-UR-92-2823.
4. Data Format For The Interchange of Fingerprint Information, American National Standard, ANSI/NBS-CLS 1-1993.
5. Digital Compression and Coding of Continuous-tone Still Images, ISO 10918-1 JPEG Draft International Standard.
6. T. Hopper, F. Preston, "Compression of Gray-Scale Fingerprint Images," in *Proc. Data Compression Conference*, Snowbird, Utah, March 24-27, 1992, pp. 309-318.
7. V. Oppenheim and R. W. Schafer, Discrete Time Signal Processing, Prentice-Hall, 1989.

Part 2 - Compliance Testing

1 Scope

Part 2 of this Specification is concerned with compliance tests for the continuous-tone gray-scale digital fingerprint image compression encoding process, the decoding process, and the compressed image data formats specified in Part 1.

This Specification specifies compliance tests for the Part 1 encoding and decoding of fingerprint images. Among the purposes of these tests is to ensure that the compressed image data format is properly implemented, and that the encoder and decoder implementations compute the SWT and quantization functions with sufficient accuracy.

This Specification:

- specifies compliance tests for the Part 1 compressed fingerprint image data formats;
- specifies compliance tests for the Part 1 encoding process used for compressing fingerprint images based on Wavelet Scalar Quantization (WSQ);
- specifies compliance tests for the Part 1 decoding process used for reconstructing WSQ compressed fingerprint images.

2 Introduction

This document (Part 2) contains the compliance tests for the WSQ gray-scale fingerprint image compression algorithm. These tests shall be employed to test compliance of compressed fingerprint image data and of WSQ encoders and decoders to the coding process defined in Part 1 of this Specification. Encoder and decoder tests are separately specified.

This document is intended to define the basic tests to be used for certifying compliance with the major features and attributes of the algorithm. Adherence to overall accuracy and data formatting requirements and flexibility shall be tested by this specification.

This document does not contain a complete and thorough set of procedures to fully test all implementations adhering to Part 1 of this Specification. Furthermore, interoperability for different implementations of WSQ encoders and decoders other than those specific implementations tested for compliance cannot be assumed.

Image compression algorithms necessitating a high compression ratio will be inherently lossy. Any loss detected in image reproduction quality will be the result of DWT calculation noise and the coefficient quantization. Accuracy requirements described in this part of the specification ensure satisfactory operation if the DWT and quantization table entries employed for compliance testing are considered appropriate.

2.1 Compressed Image Data Compliance Tests

Compliance tests described in this document may be used to determine whether a particular compressed image data set resulting from a specific implementation follows all of the syntactical rules set forth in Part 1 of this Specification. Specific tests have been defined to determine an implementation's compliance to the interchange format, the abbreviated format syntax for compressed image data, and the abbreviated format syntax for table-specification data.

2.2 Encoder Compliance Tests

The compliance tests for the encoders have been defined to be independent of any specific type of fingerprint image. Compliance tests for the DWT-based encoding process designate that the DWT coefficients generated by the encoder are computed within acceptable accuracy limits. Compliant coders shall be able to encode fingerprint image data and required tables in accordance with the interchange format, the abbreviated format for compressed image data, and the abbreviated format for table specification data.

2.3 Decoder Compliance Tests

In order to maximize compatibility and interchangeability, more requirements have been imposed on decoder functionality than on encoder functionality. Compliant decoders shall be able to accept the compressed image data interchange format, the abbreviated format for compressed image data, and the abbreviated format for table specification data.

Compliance tests for the DWT-based decoding process require that the proposed decoder accept as input the specified compressed bitstreams and produce as output, reconstructed image data within specified accuracy limits.

2.4 Compliance Test Data

Input images used for the compliance tests shall correspond in size to those areas on a fingerprint card containing the rolled image areas in a fingerbox, the plain thumb image area, and the plain four-finger image area. Input test sets shall contain a sample consisting of the fourteen images normally found on a ten-print fingerprint card. Each of the fourteen images will have a scanning resolution representation of 500 - 520 pixels per inch (ppi). This is the range of resolutions permitted in the ANSI Standard for Fingerprint Information Interchange. The samples of each test image have 8-bit precision. Table 1 lists the distribution of the fourteen component set together with the size and approximate pixel dimensions for each image type.

Image	Resolution	Size (Inches)	Number	Approx. Pixel Dim.
Rolled	520	1.500 x 1.500	10	780 x 780
Plain Thumb	520	0.875 x 1.875	2	455 x 975
Plain Four	520	3.125 x 1.875	2	1625 x 975

Table 1. Test Set Images

2.5 Range of Size Values Not Tested

To minimize the complexity and the number of compliance tests, the size parameter is not tested over its entire range. Images which have an area no greater than the limits established in the ANSI Standard for Information Interchange but with other dimensions than those listed in Table 1 shall be able to be processed by any of the compliant encoders or decoders. However, no compliance tests are specified in this document addressing images with dimensions larger than those contained in Table 1. Furthermore, compliant encoders or decoders are not required to be able to process images with either or both dimensions which are less than 400.

3 Compressed Data Format Compliance Testing

Compressed data format requirements state that any compressed image data represented in one of the compressed data formats (interchange format or abbreviated format) shall comply with the syntax and code assignments appropriate for the decoding process.

Compliance to one of the compressed data formats is determined by taking each of the constituent parts (parameters, markers, marker segments, and entropy coded segments) of the compressed image data under test and verifying that each of these parts complies with the syntax and code assignments. No step-by-step procedure for compliance testing is contained in this section. Instead a table is provided to assist in determining whether the parts of the compressed image data under test satisfy the requirements for the coding process.

Table 2 to Table 5 give specific references to syntax requirements for compressed image data. The Part 1 references in the second columns indicate where these requirements are stated.

3.1 Interchange Format Syntax Requirements

The encoder output shall comply with the interchange format syntax and code assignments.

Marker Segments	Pt 1 ref	Figure	Requirement
SOI	B.2.1	Figure B.2	Required
Frame	B.2.1	Figure B.2	Required
[tables/misc]	(See Table 3 below)		Optional
Frame Header	B.2.2	Figure B.3	Required
Parameters	B.2.2	Figure B.3	Required
First Block	B.2.1	Figure B.2	Required
.....			Optional
Last Block	B.2.1	Figure B.2	Optional
EOI	B.2.1	Figure B.2	Required

Table 2. Compressed Data Syntax Requirements

Marker Segments	Pt 1 ref	Figure	Requirement
[tables/misc]	B.2.4	Figure B.5	Required
DTT	B.2.4.1	Figure B.6	Required
Parameters	B.2.4.1	Figure B.6	Required
DQT	B.2.4.2	Figure B.7	Required
Parameters	B.2.4.2	Figure B.7	Required
DHT	B.2.4.3	Figure B.8	Required
Parameters	B.2.4.3	Figure B.8	Required
DRI	B.2.4.4	Figure B.9	Optional
Parameters	B.2.4.4	Figure B.9	Optional
COM	B.2.4.5	Figure B.10	Optional
Parameters	B.2.4.5	Figure B.8	Optional

Table 3. Tables/Misc Syntax Requirements

Marker Segments	Pt 1 ref	Figure	Requirement
[tables/misc]	(See Table 3 above)		Optional

SOB	B.2.3	Figure B.4	Required
Parameters	B.2.3	Figure B.4	Required
first ECS	B.2.1	Figure B.2	Optional
RSTx	B.2.4.4	Figure B.2	Optional
.....			
last ECS	B.2.1	Figure B.2	Required

Table 4. Block Syntax Requirements

3.2 Abbreviated Format Syntax Requirements

Abbreviated format data is required to comply with the syntax and code assignments as specified in Sections B.3 and B.4 of Part 1 of this Specification.

3.2.1 Abbreviated Format for Compressed Image Data Syntax

The compliance testing for abbreviated format compressed image data syntax is the same as the compliance testing for the interchange format syntax given in 3.1 except that some or all of the table specifications may be omitted. The table specifications are found in the DTT, DQT, and DHT marker segments. If all of the tables are removed from a marker segment, the marker and length parameter are also removed.

3.2.2 Abbreviated Format for Table Specification Data Syntax

Table 5 specifies the abbreviated format for table specification data syntax requirements.

Marker Segments	Pt 1 ref	Figure	Requirement
SOI	B.2.1	Figure B.11	Required
[tables/misc]	(See Table 4 above)		Required
EOI	B.2.1	Figure B.2	Required

Table 5. Table Specification Abbreviated Syntax Requirements

4 Encoder Compliance Tests

An encoding process converts source image data to compressed image data. To comply with this specification an encoder shall satisfy all three of the following requirements:

1) With proper accuracy convert source fingerprint image data to compressed image data which complies with the interchange format syntax specified in Annex B of Part 1;

2) Encode table-specification data associated with fingerprint image data to comply with the abbreviated format syntax specified in Annex B of Part 1; and

3) With proper accuracy convert source fingerprint image data to compressed image data which complies with the abbreviated format syntax specified in Annex B of Part 1.

In order to determine compliance of DWT-based encoders, the test procedure set forth in Annex AA.2 shall be performed. An encoder is found to be compliant if the generated test data meets the requirements on accuracy specified in Annex AA.2.

5 Decoder Compliance Tests

A decoding process converts compressed image data to reconstructed image data. To comply with this Specification, a decoder shall satisfy all three of the following requirements:

1) With proper accuracy, convert to reconstructed fingerprint image data any compressed image data, with parameters within the range specified by the ANSI Standard for the Data Format for the Interchange of Fingerprint Information (ANSI/NIST-CSL 1-1993)[4], and which complies with the interchange format specified in Annex B of Part 1 of this Specification;

2) Accept and properly store any table-specification data associated with fingerprint image data which complies with the abbreviated format syntax for table-specification data specified in Annex B of Part 1 of this Specification; and,

3) With proper accuracy, convert to reconstructed fingerprint image data any compressed fingerprint image data which complies with the abbreviated format syntax for compressed image data specified in Annex B of Part 1 of this Specification, provided that the table-specification data required for decoding the compressed image data has previously been installed in to the decoder.

In order to determine compliance of DWT-based decoders, the procedure set forth in Annex AA.3 shall be performed. A decoder is compliant if the resulting test data meets the requirements on accuracy specified in Annex AA.3.

Compressed bitstream input test data and reference test data are available from the FBI to parties who wish to validate their implementation of a DWT-based decoder for fingerprint data.

Compliance for DWT-based decoders require successful completion of Test A and Test B. Each test defines its own compressed bitstream structure. The testing procedure shall be repeated for each test using the specified compressed bitstream as test input and the output data produced by each test must satisfy the requirements for all DWT-based decoders.

The structure of the bitstreams used by the two tests are described as follows:

Test A:

Bitstream A:	Interchange format syntax

Test B:

Bitstream B1:	Abbreviated format syntax Transform, quantization, and Huffman tables No entropy coded segments
Bitstream B2 :	Abbreviated format syntax Entropy coded segments

Test A specifies a compressed bitstream which conforms to the syntax of the Interchange Format. Test B employs two bitstreams which conform to the abbreviated format syntax. The bitstreams B1 and B2 must be decoded in succession. The output test data produced after bitstream B2 is decoded must satisfy the requirements on accuracy for all DWT-based decoders.

Annex AA. Procedures for Determining Compliance

The compliance procedures defined within this specification require that the output data sets generated by the implementation under test match the output data sets from a reference implementation within the stated requirements on accuracy. This will ensure that new implementations have the proper numerical precision and interoperate with other compliant implementations. Encoders are required to implement the parameter setting specified in Part 1: FBI Parameter Setting and Decoders must handle the range of parameter values specified in Part 1: Requirements and Guidelines.

AA.2. Encoder Compliance Measures

1) The compressed file size (excluding comments) produced by the implementation under test shall be within 0.4% of the reference compressed file size.

$$\frac{S_T - S_R}{S_R} 100 \le .4$$

where S_T and S_R are the file sizes for the implementation under test and the refrence implementation respectively.

2) All quantization bin widths (including the zero bins) shall be within .051% of the corresponding bin widths contained in the quantization table within the reference compressed image.

$$\frac{Q_{k,T} - Q_{k,R}}{Q_{k,R}} 100 \le .051 \quad and \quad \frac{Z_{k,T} - Z_{k,R}}{Z_{k,R}} 100 \le .051 \qquad 0 \le k \le 59$$

where $Q_{k,R}$ and $Q_{k,T}$ are the quantization bin widths for the k^{th} subband calculated by the reference and the test implementations respectively. $Z_{k,R}$ and $Z_{k,T}$ are the corresponding zero bin widths.

3) At least 99.99% of the bin index values, $p_k(m,n)$, within the test implementation compressed image file shall be the same as the corresponding values in the reference compressed image file and no bin index value shall differ by more than 1.

AA.3. Decoder Compliance Measures

1) At least 99.9% of the test implementation reconstructed values shall be the same as the values in the reference reconstructed image and no values from the implementation under test shall differ from the reference reconstructed image by more than 1.

Part 3 – FBI Parameter Settings, Encoder Number Two

Part 3 of this Specification contains procedures for establishing the parameter table value for the first FBI-approved WSQ implementation, encoder number two.

1 Source Image Normalization

The following source image, *I(m,n)*, shall be normalized according to the specification in Annex A.1,

$$I'(m,n) = \frac{I(m,n) - M}{R}$$

where M is the image mean and $R = \frac{1}{128}\max(I_{max} - M,\ M - I_{min})$.

I_{min} and I_{max} are, respectively, the minimum and maximum pixel values in the image *I(m,n)*.

2 Transform Table

The following table contains the impulse response coefficients for the analysis filters in Encoder Number 1. This is a WS-type filter bank and is applied to the WSS signal extension, $y = E_s^{(1,1)}x$, using the SWT algorithm described in Annex A of Part 1. The synthesis filters, f_0 and f_1, are defined by the anti-aliasing relations given in Annex A.

Tap	Exact Value	Approx. Value
$h_o(0)$	$-5\sqrt{2}x_1(48\lvert x_2\rvert^2 - 16\Re x_2 + 3)/32$	0.85269867900940
$h_o(\pm1)$	$-5\sqrt{2}x_1(8\lvert x_2\rvert^2 - \Re x_2)/8$	0.37740285561265
$h_o(\pm2)$	$-5\sqrt{2}x_1(4\lvert x_2\rvert^2 + 4\Re x_2 - 1)/16$	-0.11062440441842
$h_o(\pm3)$	$-5\sqrt{2}x_1(\Re x_2)/8$	-0.02384946501938
$h_o(4\pm)$	$-5\sqrt{2}\,x_1/64$	0.037828455506995
$h_1(-1)$	$\sqrt{2}(6x_1 - 1)/16\,x_1$	0.78848561640566
$h_1(-2,0)$	$-\sqrt{2}(16x_1 - 1)/64\,x_1$	-0.41809227322221
$h_1(-3,1)$	$\sqrt{2}(2x_1 + 1)/32\,x_1$	-0.040689417609558
$h_1(-4,2)$	$-\sqrt{2}/64\,x_1$	0.064538882628938

where:

$$x_1 = A + B - 1/6$$

$$x_2 = -(A + B)/2 - 1/6 + i\sqrt{3}(A - B)/2$$

$$A = \left(\frac{-14\sqrt{15} + 63}{1080\sqrt{15}}\right)^{1/3}$$

$$B = \left(\frac{-14\sqrt{15} - 63}{1080\sqrt{15}}\right)^{1/3}$$

Table 1. Analysis Wavelet Filters

3 Adaptive Quantization of DWT Output

This section defines the parameters Q_k, Z_k, and C from Annex A.3 for WSQ encoder number two. The dequantization output level parameter, C, used in the quantization decoder shall be set to the value

$$C = 0.44$$

The bin widths, Q_k and Z_k, are determined from the variances of the DWT subbands as follows.

3.1 Subband variance computation

A subband variance computation is made based on either a subregion estimate or the entire region of each DWT subband. The decision on which region depends on the sum of the subregion variance estimates for DWT subbands 0-3. If $\sum_{k=0}^{3} \sigma_k^2 > 20{,}000$, the variance estimate is based on the subregion of each DWT subband, otherwise the variance computation uses the full region of each DWT subband.

For the variance estimate based on a subregion of each DWT, let $a_k(m,n)$ denote the floating point array of which X_k and height Y_k comprising the k^{th} subband, indexed as $0 \le m < Y_k$ and $0 \le n < X_k$ with (0,0) referring to the upper left corner of the subband. The width and height of the subregion used for the variance estimate are, respectively,

$$X'_k = \left\lfloor \frac{3X_k}{4} \right\rfloor \; and \;\; Y'_k = \left\lfloor \frac{7Y_k}{16} \right\rfloor$$

or for the variance computed on the entire region of each DWT the width and height are,

$$X'_k = X_k \; and \; Y'_k = Y_k$$

The variance shall be calculated with the unbiased estimator

$$\sigma_k^2 = \frac{1}{X'_k Y'_k - 1} \sum_{n=x_{0,k}}^{x_{1,k}} \sum_{m=y_{0,k}}^{y_{1,k}} (a_k(m,n) - \mu_k)^2$$

where μ_k denotes the mean of a_k.

The horizontal and vertical offsets for the subregions ($x_{i,j}$ and $y_{i,k}$, respectively), relative to the upper left corner, are

$$x_{0,k} = \left\lfloor \frac{X_k}{8} \right\rfloor$$

$$x_{1,k} = x_{0,k} + X'_k - 1$$

$$y_{0,k} = \left\lfloor \frac{9Y_k}{32} \right\rfloor$$

$$y_{1,k} = y_{0,k} + Y'_k - 1$$

The horizontal and vertical offsets for the full region ($x_{i,j}$ and $y_{i,k}$, respectively), relative to the upper left corner, are

$$x_{0,k} = 0$$

$$x_{1,k} = X_k - 1$$

$$y_{0,k} = 0$$

$$y_{1,k} = Y_k - 1$$

3.2 Bin width computation

The formula for the relative bin widths, Q'_k, used in encoder number two is:

$$Q'_k = qQ_k = \begin{cases} 1 & k = 0-3 \\ 10/(A_k \log_e(\sigma_k^2)) & k = 4-59 \text{ and } \sigma_k^2 \geq 1.01 \\ 0 & k = 60-63 \text{ or } \sigma_k^2 < 1.01 \end{cases} \qquad A_k = \begin{cases} 1.32 & k = 52,56 \\ 1.08 & k = 53,58 \\ 1.42 & k = 54,57 \\ 1.08 & k = 55,59 \\ 1.00 & \text{otherwise} \end{cases}$$

The proportionality constant, q, controls the absolute bin widths Q_k, and the overall level of compression. Zero bin widths, Z_k, shall be computed in terms of Q_k by the formula $Z_k = 1.2Q_k$.

We now specify the procedure for computing the parameter q that determines the bin widths Q_k as specified above. For the k^{th} DWT subband, let σ_k^2 denote the subband variance estimate, computed according to the above specification. Let m_k be the downsample factor, which is defined to be the ratio of image size to subband size; e.g., $m_{63} = 16$ and $m_4 = 256$. The bit rate to be assigned to the k^{th} DWT subband will be denoted r_k, and r is the targeted overall bit rate, which imposes a constraint on the subband bit rates via the relation

$$r = \sum_k \frac{r_k}{m_k}$$

As explained in Annex A.3, the standard allows the decoder to discard some subbands and transmit a bin width of zero ($Q_k = 0$) to signify that no compressed image data is being transmitted for subband k. For instance, this is always done for $60 \leq k \leq 63$ in encoder number two, and may be done for other subbands as well on an image by image basis if the encoder determines that a certain subband contains so little information that it should be discarded altogether. To keep track of the subband bit allocateion, let K denote the set of all subbands assigned positive bit rates (in particular, for encoder number two, $K \subset \{0, 1, ..., 59\}$). The fraction of DWT coefficients being coded at a positive bit rate will be denoted by S, where

$$S = \sum_{k \in K} \frac{1}{m_k}$$

To relate bit rates to quantizer bin widths, we model the data in each subband as lying in some interval of finite extent, specifically, as being contained within an interval spanning 5 standard deviations. This assumption may not be valid in general, but we will not incur overload distortion due to outliers because outliers are coded using escape sequences in the Huffman coding model. Therefore, for the sake of quantizer design we assume that the data lies in the interval $[\mu_k - \gamma\sigma_k, \ \mu_k + \gamma\sigma_k]$; this implies that

$$Q_k = \frac{2\gamma\sigma_k}{L_k}$$

where L_k is the number of bins in the quantizer, and the loading factor, γ , has the value 2.5. We model the average transmission bit rate for subband k by

$$r_k = \log_2 L_k \text{ bits/sample.}$$

The formula for q, in terms of the parameters given above, is then

$$q = \gamma^{-1} 2^{r/S-1} \left[\prod_{k \in K} \left(\frac{\sigma_k}{Q'_k} \right)^{1/m_k} \right]^{-1/S}$$

Two cases require special attention. First, to prevent overflow in Q'_k if $\log_e(\sigma_k^2) \approx 0$, the encoder shall discard any subband for which $\sigma_k^2 < 1.01$ by setting $Q_k = 0$. Second, if $Q_k > 2\gamma\sigma_k$ then the above quantization model implies that $r_k < 0$. Since this is not physically meaningful, we use an iterative procedure (specified below) to determine q. The iterative procedure excludes from the bit allocation those subbands that have theoretically nonpositive bit rates; this will ensure that the overall bit rate constraint, r, is met. Once the bin widths have been determined, the quantizers defined in Annex A.3 shall be applied to each subband for which a bin width has been computed, including those bands with theoretically nonpositive bit rates.

NOTE: For subbands with theoretically nonpositive bit rates (i.e., the bands listed in the set $K^{(0)} \backslash K$ determined below), we encode anyway using the (large) bin widths given by the bin with specification in the expectation that quantization of these bands will effectively result in zero subband bit rates. It may nonetheless happen that a few samples in such bands actually get mapped to nonzero values by quantization and therefore contribute information to the reconstructed image.

Iterative Procedure for Computing Bin Widths

1. Initialize:

   (a) $j = 0$;

   (b) $K^{(0)} = \left\{ k | 0 \leq k \leq 59 \; and \; \sigma_k^2 \geq 1.01 \right\}$

2. Iterate on j to calculate q:

   (a)

$$S^{(j)} = \sum_{k \in K^{(j)}} \frac{1}{m_k} \quad ;$$

   (b)

$$q^{(j)} = \gamma^{-1} 2^{r/S^{(j)}-1} \left[\prod_{k \in K^{(j)}} \left(\frac{\sigma_k}{Q'_k} \right)^{1/m_k} \right]^{-1/S^{(j)}}$$

3. Exclude bands that would contribute theoretically nonpositive bit rates:

   (a) $\Xi^{(j)} = \left\{ k \in K^{(j)} \middle| Q'_k \,/\, q^{(j)} \geq 2\gamma\sigma_k \right\}$

   (b) If $\Xi^{(j)} \neq 0 \; then$

      i. $K^{(j+1)} = K^{(j)} \setminus \Xi^{(j)}$

      ii. $j = j + 1$

      iii. go to step 2;

   else

      i. $q = q^{(j)}$

      ii. $K = K^{(j)}$

      iii. continue.

4. Calculate bin Widths:

   $If \; k \in K^{(0)} \; then$

   $Q_k = Q'_k / q$

   else

   $Q_k = 0$

5. Exit.

The backslash \ denotes the set difference operator; i.e., $A \setminus B = A \cap B^c$

4 Huffman Coding of Quantizer Indices

The subbands produced by encoder number two shall be divided into 3 blocks for Huffman coding, with one Huffman encoder constructed for block 1 (subbands 0 through 18) and a second Huffman encoder constructed for use on both blocks 2 and 3 (subbands 19-51 and 52-59, respectively). Both Huffman encoders shall construct Huffman codes for nonzero quantizer indices between -73 and +74, inclusive, and zero run lengths from 1 to 100 as specified in section A.4.1. All symbols outside this range shall be coded with escape sequences.

NOTE: Recall from the specification of the Huffman coding model in Annex A.4.1 that symbol number 180, corresponding to the quantizer index value 0, is never used. An isolated zero shall always be coded as a zero run of length 1 using symbol number 1.