NOT MEASUREMENT SENSITIVE

NOTE: The cover page of this standard has been changed for administrative reasons. There are no other changes to this document.

MIL-STD-188-198A
15 December 1993
SUPERSEDING
MIL-STD-188-198
18 June 1993

DEPARTMENT OF DEFENSE
INTERFACE STANDARD

JOINT PHOTOGRAPHIC EXPERTS GROUP (JPEG)
IMAGE COMPRESSION
FOR THE
NATIONAL IMAGERY TRANSMISSION FORMAT STANDARD

AMSC N/A

AREA TCSS

DISTRIBUTION STATEMENT A. Approved for public release; distribution is unlimited.

FOREWORD

1. The National Imagery Transmission Format Standard (NITFS) is the standard for formatting digital imagery and imagery-related products and exchanging them among members of the Intelligence Community (IC) as defined by Executive Order 12333, the Department of Defense (DOD), and other departments and agencies of the United States Government, as governed by Memoranda of Agreement (MOA) with those departments and agencies.

2. The National Imagery Transmission Format Standard Technical Board (NTB) developed this standard based upon currently available technical information.

3. The DOD and members of the Intelligence Community are committed to interoperability of systems used for formatting, transmitting, receiving, and processing imagery and imagery-related information. This standard describes the Joint Photographic Experts Group (JPEG) compression algorithm and establishes its application within the NITFS.

4. Beneficial comments (recommendations, additions, deletions) and other pertinent data which may be of use in improving this document should be addressed to Defense Information Systems Agency (DISA), Joint Interoperability and Engineering Organization (JIEO), Center for Standards (CFS), Attn: TBCF, 11440 Isaac Newton Square, North, Reston, VA 22090 by using the Standardization Document Improvement Proposal (DD Form 1426) appearing at the end of this document or by letter.

5. Portions of this document were reproduced from the Draft CCITT Recommendation T.81 (1992) with prior authorization from the copyright holder, International Telecommunications Union (ITU). DOD takes sole responsibility for the selection of portions from the ITU document for reproduction, with subsequent additions or changes, which should not be attributed to the ITU.

CONTENTS

CONTENTS

CONTENTS

CONTENTS

CONTENTS

1. SCOPE

1.1 Scope. This standard establishes the requirements to be met by systems complying with NITFS when image data are compressed using the JPEG image compression algorithm as described in DIS 10918-1, *Digital Compression and Coding of Continuous-tone Still Images*.

1.2 Content. This standard provides technical detail of the NITFS compression algorithm designated by the code C3 in the Image Compression field of the National Imagery Transmission Format (NITF) file image subheader, JPEG, for both eight- and 12-bit gray scale imagery and 24-bit color imagery. It also provides the required default quantization tables for use in Secondary Imagery Dissemination Systems (SIDS) complying with NITFS.

1.3 Applicability. This standard is applicable to the Intelligence Community and the Department of Defense. It is mandatory for all Secondary Imagery Dissemination Systems in accordance with the memorandum by the Assistant Secretary of Defense for Command, Control, Communications, and Intelligence ASD(C^3I) Subject: National Imagery Transmission Format Standard (NITFS), 12 August 1991. This standard shall be implemented in accordance with the Joint Interoperability and Engineering Organization (JIEO) Circular 9008 and MIL-HDBK-1300. New equipment and systems, those undergoing major modification, or those capable of rehabilitation shall conform to this standard.

1.4 Tailoring task, method, or requirement specifications. The minimum compliance requirements for implementation of this compression algorithm are defined in JIEO Circular 9008.

1.5 Types of operation. This standard establishes the requirements for the communication or storage for interchange of image data in compressed form. Each type of operation defined by this standard consists of three parts:

a. The compressed data interchange format (which defines the image data field of the NITF file format).

b. The encoder.

c. The decoder.

This standard currently defines three types of operation:

a. Type 1 - 8-bit sample precision gray scale sequential Discrete Cosine Transform (DCT) with Huffman coding.

b. Type 2 - 24-bit color, 8-bit sample precision per component, sequential DCT with Huffman coding.

c. Type 3 - 12-bit sample precision gray scale sequential DCT with Huffman coding.

d. Types 4-N - To Be Revised (TBR).

Additional types of conforming JPEG encoding methods and extensions to the conforming interchange format are expected to be added in future versions of this standard as soon as technical work codifies their requirements and validates fitness for use.

2. APPLICABLE DOCUMENTS

2.1 Government documents.

2.1.1 Specifications, standards, and handbooks. The following specifications, standards, and handbooks form a part of this document to the extent specified herein. Unless otherwise specified, the issues of these documents are those listed in the issue of the Department of Defense Index of Specifications and Standards (DODISS) and supplements thereto, cited in the solicitation.

STANDARDS

FEDERAL

FED-STD-1037B	-	Telecommunications: Glossary of Telecommunication Terms, 3 June 1991.

MILITARY

MIL-STD-2500	-	National Imagery Transmission Format (NITF) for the National Imagery Transmission Format Standard (NITFS).

HANDBOOKS

MIL-HDBK-1300	-	National Imagery Transmission Format Standard (NITFS).

(Unless otherwise indicated, copies of federal and military specifications, standards, and handbooks are available from the Standardization Documents Order Desk, 700 Robbins Avenue, Building #4, Section D, Philadelphia, PA 19111-5094.)

2.1.2 Other Government documents, drawings, and publications. The following other Government documents form a part of this document to the extent specified herein. Unless otherwise specified, the issues of these documents are those cited in the solicitation.

DISA/JIEO Circular 9008	-	NITFS Certification Test and Evaluation Program Plan, (Effectivity 8).

(Copies of DISA/JIEO Circular 9008 may be obtained from DISA/JIEO/JITC/TCBD, Fort Huachuca, AZ 85613-7020.)

2.2 Non-Government publications. The following documents form a part of this document to the extent specified herein. Unless otherwise specified, the issues of the documents which are adopted by (DOD) are those listed in the issue of the DODISS cited in the solicitation.

International Telecommunication Union (ITU) / INTERNATIONAL TELEGRAPH AND TELEPHONE CONSULTATIVE COMMITTEE (CCITT)

Draft CCITT Recommendation T.81 ISO/IEC 10918-1	-	Information Technology - Digital Compression and Coding of Continuous-Tone Still Images. Part I: Requirements and Guidelines, September 1992.

INTERNATIONAL RADIO CONSULTATIVE COMMITTEE

CCIR Recommendation 601-1	-	Encoding Parameters of Digital Television for Studios.

(Copies may be obtained from ANSI, 11 W. 42nd Street, New York, NY 10036, Attn: JPEG Draft Specification. Non-government standards and other publications are normally available from the organizations that prepare or distribute the documents. These documents also may be available in or through libraries or other informational services.)

2.3 Order of precedence. In the event of a conflict between the text of this standard and the references cited herein, the text of this standard shall take precedence. Nothing in this standard, however, shall supersede applicable laws and regulations unless a specific exemption has been obtained.

3. DEFINITIONS

3.1 Acronyms used in this standard. The following definitions are applicable for the purpose of this standard. In addition, terms used in this standard and defined in the FED-STD-1037B shall use the FED-STD-1037B definition unless noted.

a.	ASD(C^3I)	Assistant Secretary of Defense for Command, Control, Communications, and Intelligence
b.	CFS	Center for Standards
c.	DCT	Discrete Cosine Transform
d.	DISA	Defense Information Systems Agency
e.	DOD	Department of Defense
f.	DODISS	Department of Defense Index of Specifications and Standards
g.	FDCT	Forward Discrete Cosine Transform
h.	IC	(1) Intelligence Community (2) Image Compression
i.	IDCT	Inverse Discrete Cosine Transform
j.	JIEO	Joint Interoperability and Engineering Organization (formerly JTC^3A)
k.	JPEG	Joint Photographic Experts Group
l.	LSB	Least Significant Bit
m.	MCU	Minimum Coded Unit
n.	MOA	Memorandum of Agreement
o.	MSB	Most Significant Bit
p.	NITF	National Imagery Transmission Format
q.	NITFS	National Imagery Transmission Format Standard

r.	NTB	National Imagery Transmission Format Standard Technical Board
s.	RGB	Red, Green, Blue
t.	SIDS	Secondary Imagery Dissemination System
u.	SLIP	Serial Line Internet Protocol
v.	TBR	To Be Revised
w.	YCbCr601	Y = Brightness of signal, Cb = Chrominance (blue), Cr = Chrominance (red).

3.2 Definitions used in this standard. The definitions used in this document are defined as follows:

a. Abbreviated format - A representation of compressed image data that is missing some or all of the table specifications required for decoding.

b. AC coefficient - Any Discrete Cosine Transform (DCT) coefficient for which the frequency is not zero in at least one dimension.

c. Arithmetic decoder - An embodiment of an arithmetic decoding procedure.

d. Arithmetic decoding - An entropy decoding procedure that recovers the sequence of symbols from the sequence of bits produced by the arithmetic encoder.

e. Arithmetic encoder - An embodiment of an arithmetic encoding procedure.

f. Arithmetic encoding - An entropy encoding procedure that codes by means of a recursive subdivision of the probability of the sequence of symbols coded up to that point.

g. Baseline (sequential) - A particular sequential DCT-based encoding and decoding process specified in this standard; required for all DCT-based decoding processes.

h. Bit stream - A bit-stream is a sequence of binary digits. The term is principally used to describe data that is being moved internally within a computer, or being transferred between computers. For the purpose of MIL-STD-188-198 (JPEG), a bit stream is defined as a partially encoded or decoded sequence of bits comprising an entropy-coded segment.

i. Block-row - A sequence of eight contiguous component lines that are partitioned into 8x8 blocks.

j. Byte - A sequence of N adjacent binary digits, usually treated as a unit, where N is a non zero integral number. Note: In pre-1970 literature, "byte" referred to a variable length field.

k. Byte stuffing - A procedure in which either the Huffman coder or the arithmetic coder inserts a zero byte into the entropy-coded segment following the generation of an encoded hexadecimal 0xFF byte. For the purpose of NITFS, in Serial Line Internet Portocol (SLIP), a technique used to avoid spurious appearances of the END character within a frame.

l. C3 - The code used to indicate the JPEG compression algorithm in the image compression (IC) field of the image subheader.

m. Chromaticity - Property of a color stimulus defined by its chromacity coordinates (for NITFS, use YCbCr601 chromaticity coordinates).

n. Chrominance - 1. Perceptual color attribute consisting of the hue and saturation of a color. 2. The difference determined by quantitative measurement between a color and a chosen reference color of the same luminous intensity, the reference color having a specified color quality. 3. The quality of the color without reference to brightness.

o. Class (of coding process) - Lossy or lossless coding processes.

p. Coding model - A procedure used to convert input data into symbols to be coded.

q. Coding process - A general term for referring to an encoding process, a decoding process, or both.

r. Color image - A continuous-tone image that has more than one colorimetry component.

s. Columns - Samples per line in a component.

t. Component - For the NITFS, one of the two-dimensional arrays that comprise an image. Used interchangeably with band.

u. Compressed data - Either compressed image data, or table specification data, or both.

v. Compressed image data - A coded representation of an image, as specified in MIL-STD-188-198 (JPEG).

w. Compression - For the NITFS, reduction in the number of bits used to represent source image data.

x. COMRAT - The compression rate code field in the NITF image subheader used to indicate the quantization matrices used.

y. Continuous-tone image - An image whose components have more than one bit per sample.

z. Data unit - A block in DCT-based processes; a sample in lossless processes.

aa. DC coefficient - The DC coefficient for which the frequency is zero in both dimensions. Note: The DC coefficient (in the context of JPEG compression) is a measure of the average value of the 64 image samples within an 8x8 block. Because of the usually strong correlation between the DC coefficients of adjacent blocks, the DC coefficients can be very efficiently encoded, and are treated separately from the encoding of the AC coefficients.

ab. DC prediction - The procedure used by DCT-based encoders whereby the quantized DC coefficient from the previously encoded 8x8 block of the same component is subtracted from the current quantized DC coefficient.

ac. DCT coefficient - The amplitude of a specific cosine basis function. Note: The Discrete Cosine Transform changes the representation of an image from a set of numbers representing the brightness of each pixel to another set of numbers that can be used to reconstruct the image mathematically. The process is similar to synthesizing music electronically from separate tones. In this case the "tones" are cosine basis functions, each of which have the properties of amplitude and frequency; but instead of referring to amplitude and frequency as a function of time, these properties relate to each of the two principal directions across the image. For this reason, the term spatial frequency is often used to emphasize that the process involves direction rather than time. The coefficients in the DCT matrix are the amplitudes of these basis functions. There is always one basis function with a zero frequency in both directions. In simple terms, "zero frequency" implies a constant, and the coefficient of this wave is labeled DC. All other coefficients have at least one non-zero directional component, and are labeled AC. The terms DC and AC are analogous to the zero-frequency (DC) and non-zero frequency (AC) components in electrical circuits.

ad. Decoder - An embodiment of a decoding process.

ae. Decoding process - For the NITFS, a process that takes compressed image data as its inputs and outputs a continuous-tone image.

af. Dequantization - The inverse procedure to quantization by which the decoder recovers a representation of the DCT coefficients.

ag. (Digital) reconstructed image (data) - A continuous-tone image which is the output of any decoder defined in this standard.

ah. (Digital) source image (data) - A continuous-tone image used as input to any encoder defined in this standard.

ai. (Digital) (still) image - A set of two-dimensional arrays of data.

aj. Discrete Cosine Transform (DCT) - Either the forward discrete cosine transform or the inverse discrete cosine transform.

ak. Effectivity - Some of the capabilities specified in this document are not required as of the issue date of the document. All such capabilities are marked with effectivity numbers, (for example, Effectivity 1). Each effectivity number will be replaced by a specific date in subsequent releases of this document.

al. Encoder - An embodiment of an encoding process.

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

an. Entropy - The lower bound on the number of bits required to encode the output of a source (of information).

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

ap. Entropy decoder - A device that processes an encoded data stream to extract the original symbols with no loss of information. The device may be implemented in hardware or software.

aq. Entropy encoding - A lossless procedure that converts a sequence of input symbols into a sequence of bits so that the average number of bits-per-symbol approaches the entropy of the input symbols.

ar. Extended (DCT-based) process - A descriptive term for DCT-based encoding and decoding processes in which additional capabilities are added to the baseline sequential process.

as. Forward Discrete Cosine Transform (FDCT) - A mathematical transformation using cosine-based functions that convert a block of samples into a corresponding array of basis function amplitudes.

at. Frame - 1. For the MIL-STD-2045-44500 (TACO2), in data transmission, a sequence of contiguous bits bracketed by and including uniquely recognizable delimiters. 2. For the MIL-STD-188-198 (JPEG), a group of one or more scans (all using the same DCT-based or lossless process) through the data of one or more of an image.

au. Frame header - The start-of-frame marker and frame parameters coded at the beginning of a frame.

av. Frequency - For the purpose of this standard, a two-dimensional index into the two-dimensional array of DCT coefficients.

aw. Gray scale - An optical pattern consisting of discrete steps or shades of gray between black and white.

ax. Hierarchical - A mode of operation for coding an image in which the first frame for a given component is followed by frames that code the differences between the source data and the reconstructed data from the previous frame for that component. Resolution changes are allowed between frames.

ay. Hierarchical decoder - A sequence of decoder processes in which the first frame for each component is followed by frames that decode an array of differences for each component and adds it to the reconstructed data from the preceding frame for that component.

az. Hierarchical encoder - The mode of operation in which the first frame for each component is followed by frames that encoded the array of differences between the source data and the reconstructed data from the preceding frame for that component.

ba. Huffman decoder - An embodiment of a Huffman decoding procedure.

bb. Huffman decoding - An entropy decoding procedure that recovers the symbol from each variable length code produced by the Huffman encoder.

bc. Huffman encoder - An embodiment of a Huffman encoding procedure.

bd. Huffman encoding - An entropy encoding procedure that assigns a variable length code to each input symbol.

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

bf. Image data - Either source image data or reconstructed image data.

bg. IMODE - A field in the NITF image subheader used to indicate whether the image bands are transmitted sequentially or interleaved (by block or pixel).

bh. Interchange format - The representation of compressed image data for exchange between application environments.

bi. Interleaved - The descriptive term applied to the repetitive multiplexing of small groups of data units from each component in a scan in a specific order.

bj. Inverse discrete cosine transform - A mathematical transformation using cosine base functions that convert an array of basis function amplitudes into a corresponding block of samples.

bk. IREP - A field in the NITF image subheader used to indicate the color space (for example, YCbCr601) for compression.

bl. IREPBAND - A field in the NITF image subheader used to identify a component of the color space (for example, Y or Cb or Cr).

bm. Level shift - A procedure used by DCT-based encoders and decoders, whereby each input sample either is converted from an unsigned representation to a two's complement representation or from a two's complement representation to an unsigned representation.

bn. 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).

bo. Lossless coding - The mode of operation that refers to any one of the coding processes defined in this standard in which all of the procedures are lossless.

bp. Lossy - A descriptive term for encoding and decoding processes which are not lossless.

bq. Luminance - 1. The monochromatic signal used to convey brightness information. 2. In a given direction, at a given point in the path of a beam, the luminous intensity per unit projected area.

br. Marker - A two-byte code in which the first byte is hexadecimal FF (0xFF) and the second byte is a value between 1 and hexadecimal FE (0xFE).

bs. Marker segment - A marker and associated set of parameters.

bt. Minimum coded unit - The smallest group of data units that is coded.

bu. Modes (of operation) - The four main categories of image compression processes defined in this standard.

bv. Non-differential frame - The first frame for any components in a hierarchical encoder or decoder. The components are encoded or decoded without subtraction from reference components. The term refers also to any frame in modes other than the hierarchical mode.

bw. Non-interleaved - The descriptive term applied to the data unit processing sequence when the scan has only one component.

bx. Parameters - Fixed length integers four, eight or 16 bits in length, used in the compressed data formats.

by. Point transform - Scaling of a sample or DCT coefficient.

bz. Precision - For the NITFS, the number of bits allocated to a particular sample or DCT coefficient.

ca. Predictor - A linear combination of previously encoded reconstructed values (in lossless mode coding).

cb. Procedure - A set of steps that accomplishes one of the tasks which comprises an encoding or decoding process.

cc. Progressive (coding) - One of the DCT-based or hierarchical processes defined in this standard in which each scan typically improves the quality of the reconstructed image.

cd. Progressive DCT-based - The mode of operation that refers to any one of the processes defined in 5.3 of this standard.

ce. Quantization table - The set of 64 quantization values used to quantize the DCT coefficients.

cf. Quantization value - An integer value used in the quantization procedure.

cg. Quantize - The act of performing the quantization procedure for a DCT coefficient.

ch. Restart interval - The integer number of Minimum Coded Units (MCUs) processed as an independent sequence within a scan.

ci. Restart marker - The marker that separates two restart intervals in a scan.

cj. Run (length) - Number of consecutive symbols of the same value.

ck. Sample - For the NITFS, one element in the two-dimensional array that comprises a band of the image.

cl. Scan - For the purpose of this standard, a single pass through the data for one or more of the components in an image.

cm. Scan header - The start-of-scan marker and scan parameters that are coded at the beginning of a scan.

cn. Sequential (coding) - One of the lossless or DCT-based coding processes defined in this standard in which each component of the image is encoded within a single scan.

co. Sequential DCT-based - The mode of operation which refers to any one of the processes defined in 5.2 of this standard.

cp. Table specification data - The coded representation from which the tables used in the encoder and decoder are generated.

cq. (Uniform) quantization - For the purpose of this standard, the procedure by which DCT coefficients are linearly scaled in order to achieve compression.

cr. Zig-zag sequence - A specific sequential ordering of the DCT coefficients from (approximately) lowest spatial frequency to highest.

cs. (8x8) block - An 8x8 array of samples.

ct. 3-sample predictor - A linear combination of the three nearest neighbor reconstructed samples to the left and above (in lossless mode coding).

3.3 Symbols used in this standard. The symbols used in this standard are defined as follows:

a.	AC	AC DCT coefficient
b.	APP_n	application marker
c.	BITS	16 byte list containing number of Huffman codes of each length
d.	C_u	horizontal frequency dependent scaling factor in DCT
e.	C_v	vertical frequency dependent scaling factor in DCT
f.	CAT	size category of DC difference or AC coefficient amplitude
g.	CODE	Huffman code value
h.	CODESIZE (V)	code size for symbol V
i.	COM	comment marker
j.	DC	DC DCT coefficient
k.	DCi	DC coefficient for i^{th} block in component
l.	DHT	define-Huffman-tables marker
m.	DIFF	difference between quantized DC and prediction
n.	DNL	define number of lines marker
o.	DQT	define-quantization-table marker
p.	DRI	define restart interval marker

q.	EHUFCO	Huffman code table for encoder
r.	EHUFSI	encoder table of Huffman code sizes
s.	EOB	end-of-block for sequential; end-of-band for progressive
t.	EOI	end-of-image marker
u.	FREQ(V)	frequency of occurrence of symbol V
v.	HUFFCODE	list of Huffman codes corresponding to lengths in HUFFSIZE
w.	HUFFSIZE	list of code lengths
x.	HUFFVAL	list of values assigned to each Huffman code
y.	LASTK	largest value of K
z.	m	modulo eight counter for RSTm marker
aa.	MAXCODE	table with maximum value of Huffman code for each code length
ab.	MINCODE	table with minimum value of Huffman code for each code length
ac.	OTHERS(V)	index to next symbol in chain
ad.	P	sample precision
ae.	PRED	quantized DC coefficient from the most recently coded block of the component
af.	Q1-Q5	Five quality levels with associated default quantization tables for DCT based coding
ag.	Q_{vu}	quantization value for DCT coefficient Svu
ah.	Q_{00}	quantizer value for DC coefficient
ai.	r_{yx}	reconstructed image sample
aj.	R_{vu}	dequantized DCT coefficient
ak.	RSTm	restart marker

al.	RUN	length of run of zero amplitude AC coefficients
am.	s_{yx}	sample from horizontal position x, vertical position y in block
an.	S_{vu}	DCT coefficient at horizontal frequency u, vertical frequency v
ao.	SI	Huffman code size
ap.	SOI	start-of-image marker
aq.	SOF_0	baseline DCT process frame marker
ar.	SOF_1	extended sequential DCT frame marker, Huffman coding
as.	SOS	start-of-scan marker
at.	Sq_{vu}	quantized DCT coefficient
au.	V	symbol or value being either encoded or decoded
av.	VALPTR	list of indices for 1st value in HUFFVAL for each code length
aw.	V1	symbol value
ax.	V2	symbol value
ay.	ZRL	value in HUFFVAL assigned to run of 16 zero coefficients
az.	ZZ_k	k^{th} element in zigzag sequence of DCT coefficients
ba.	ZZ_0	quantized DC coefficient in zig-zag sequence

4. GENERAL REQUIREMENTS

4.1 Interoperability. The requirements specified in this standard are intended to enable the interchange of 8- and 12-bit gray scale imagery and 24-bit color imagery compressed with JPEG. The Type 1 operation (8-bit sample precision gray scale sequential DCT with Huffman coding) is the only compression currently defined. Other modes are TBR.

4.2 Encoder. "An encoder is an embodiment of an encoding process." As shown on figure 1 [1], an encoder takes as input digital source image data and table specifications and, using a specified set of procedures, generates compressed image data that is stored in the image data field of the NITF file as shown on figure 2.

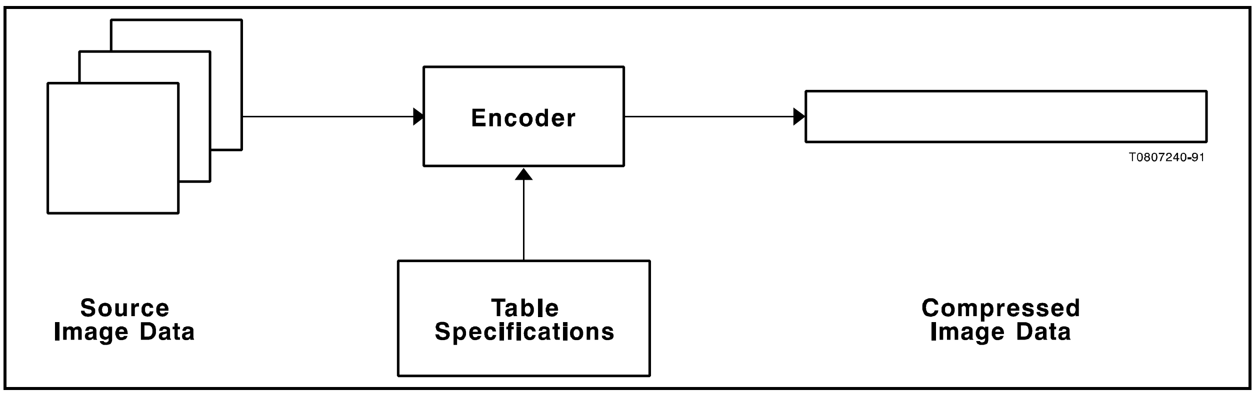

FIGURE 1. Encoder.

[1] International Telecommunication Union (ITU), CCITT Recommendation T.81 International Organization for Standardization and the International Electrotechnical Commission (ISO/IEC) 10918-1, *Information Technology - Digital Compression and Coding of Continuous-Tone Still Images, Part 1: Requirements and Guidelines* (1992), p. 17.
©Copyright by the International Telecommunication Union, 1992. All rights reserved.
Extracts from CCITT Recommendation T.81 are reproduced with the permission of the International Telecommunication Union (ITU). Selection of the extracts included in this Military Standard is the sole responsibility of the Department of Defense and is not attributable to the ITU. The complete CCITT Recommendation can be obtained from the ITU General Secretariat - Sales Section, Place des Nations, CII-1211 GENEVA 20 (Switzerland).
©Copyright by the International Organization for Standardization and the International Electrotechnical Commission, 1992. All rights reserved.
Extracts from ISO/IEC 10918-1 are reproduced with the permission of the International Organization for Standardization, ISO and the International Electrotechnical Commission, IEC. The complete standard can be obtained from any ISO or IEC member or from the ISO or IEC Central offices, Case Postale, 1211 Geneva 20, Switzerland.

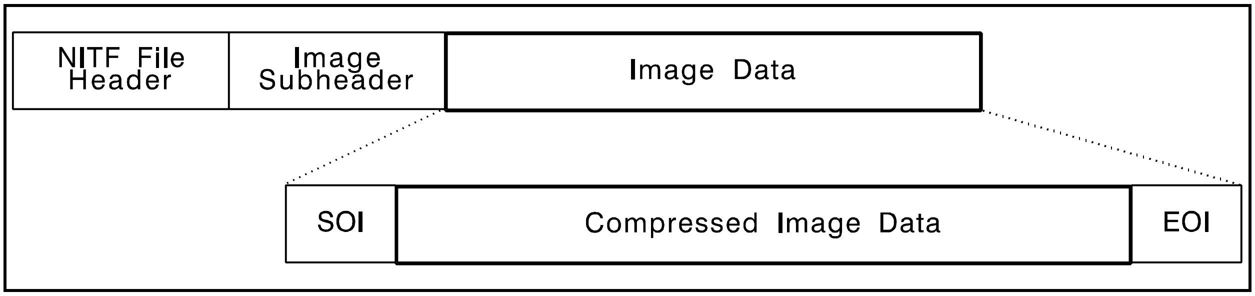

FIGURE 2. NITF file structure.

4.3 Decoder. A decoder is an embodiment of a decoding process. As shown on figure 3, a decoder takes as input compressed image data and table specifications and, by means of a specified set of procedures, generates digital reconstructed image data as output. [2]

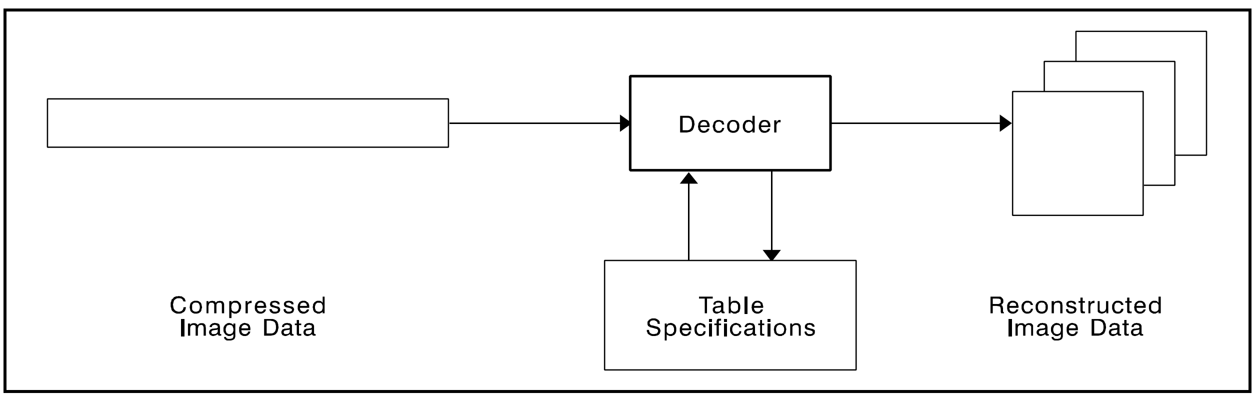

FIGURE 3. Decoder.

4.4 Interchange format-encoders. Encoders shall output to the image data field of the NITF file either a full interchange format that includes the compressed image data and all table specifications used in the encoding process, or an abbreviated format that uses default tables. The abbreviated format is identical to the full interchange format, except that it does not contain all tables required for decoding. These default tables are specified elsewhere in the NITF image subheader.

4.5 Interchange format-decoders. All decoders shall interpret full and abbreviated interchange formats.

[2] Ibid.

4.6 Lossy and lossless compression. This standard specifies two classes of encoding and decoding processes, lossy and lossless processes. Those based on the DCT are lossy, allowing substantial compression while producing a reconstructed image with high visual fidelity to the encoder's source image. The second class of coding processes is not based on the DCT and is provided for applications requiring lossless compression. These lossless encoding and decoding processes are used independently of any of the DCT-based processes.[3]

4.7 Amount of compression. The amount of compression provided by any of the various processes depends on the characteristics of the particular image being compressed and on the picture quality desired by the application.

4.8 Color conversion. Two alternative color spaces are specified for coding RGB imagery (Type 2). The first alternative is to code the RGB components directly, and the second is to transform from RGB into a luminance (Y) chrominance (Cb, Cr) color space, which is more efficient for compressing natural scenes. Since JPEG is color-blind by design, this color space must be identified in the NITF image subheader instead of the image data of the NITF file.

4.9 DCT-based coding. Figure 4 [4] shows the main procedures for all encoding processes based on the DCT. It illustrates the special case of a single-(image)block, single-component (gray scale) image. This is an appropriate simplification for overview purposes, because all image blocks are coded independently and, for each block, the processes specified in this standard, other than the color conversion, operate on each component independently.

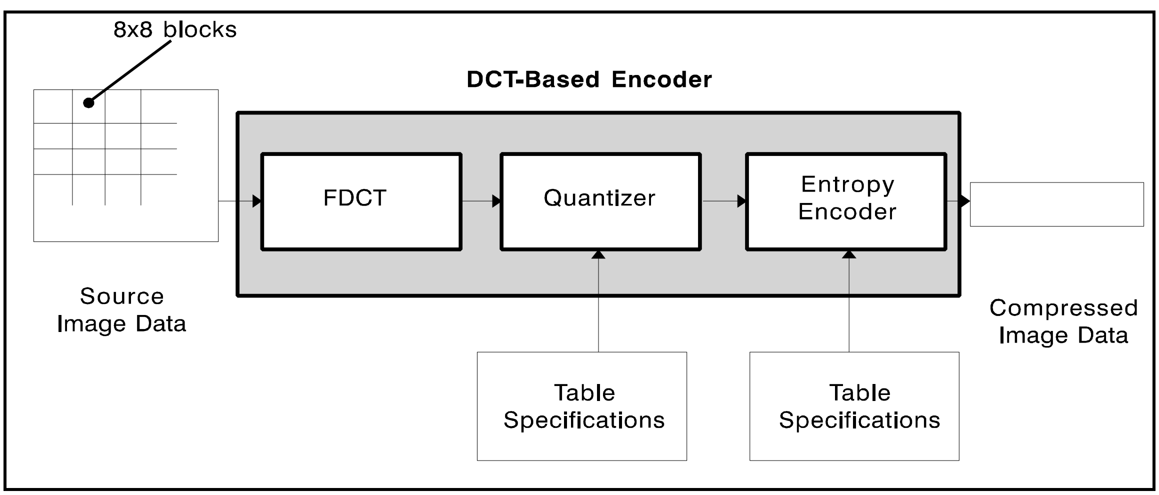

FIGURE 4. DCT-based encoder simplified diagram.

[3]Ibid, p. 18.

[4]Ibid, p. 19.

4.9.1 Forward DCT. In the encoding process, the input component's samples are grouped into 8x8 blocks, and each block is transformed by the forward DCT (FDCT) into a set of 64 values referred to as DCT coefficients. One of these values is the DC coefficient and the other 63 are the AC coefficients.

4.9.2 Quantization. Each of the 64 coefficients is using quantized one of 64 corresponding values from a quantization table (determined by one of the table specifications shown on figure 4). Unlike the ISO/CCITT JPEG standard, default quantization tables are specified in this standard.

4.9.3 Preparation for entropy coding. After quantization, the DC coefficient and the 63 AC coefficients are prepared for entropy coding, as shown on figure 5 [5]. The previous quantized DC coefficient is used to predict the current quantized DC coefficient, and the difference is encoded. The 63 quantized AC coefficients undergo no such differential encoding, but are converted into a one-dimensional zig-zag sequence, as shown on figure 5.

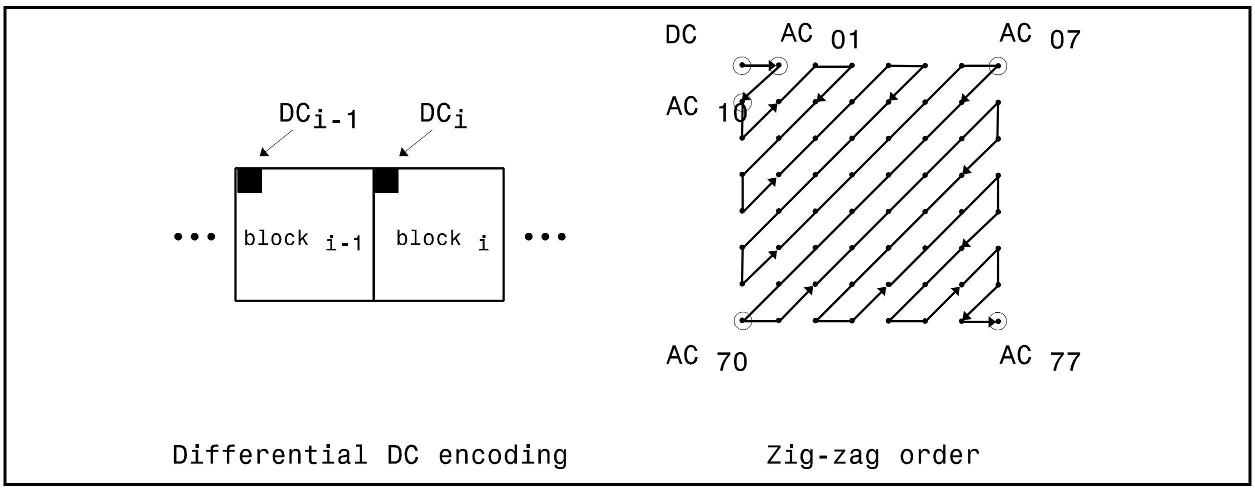

FIGURE 5. Preparation of quantized coefficients for entropy encoding.

[5] Ibid, p. 20.

4.9.4 Entropy coding. The quantized coefficients are passed to an entropy encoding procedure that compresses the data further. One of two entropy coding procedures can be used, as described in 4.12. If Huffman encoding is used, Huffman table specifications must be specified to the encoder. If arithmetic encoding is used, arithmetic coding conditioning table specifications must be provided. Unlike the ISO/CCITT JPEG standard, default Huffman tables are specified in this standard.

4.9.5 DCT-based decoding. Figure 6 [6] shows the main procedures for all DCT-based decoding processes. Each step shown performs essentially the inverse of its corresponding main procedure within the encoder. The entropy decoder decodes the zig-zag sequence of quantized DCT coefficients. After dequantization, the DCT coefficients are transformed to an 8x8 block of samples by the inverse DCT (IDCT).

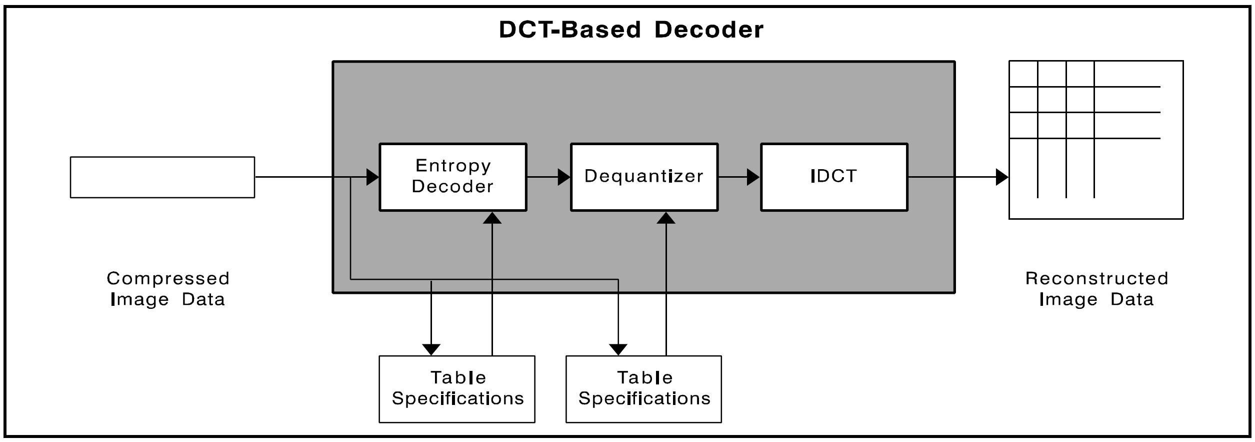

FIGURE 6. DCT-based decoder simplified diagram.

[6] Ibid.

4.10 Lossless coding. (Effectivity 2) "Figure 7 shows the main procedures for the lossless encoding processes. A predictor combines the reconstructed values of up to three neighborhood samples at positions a, b, and c to form a prediction of the sample at position x as shown in figure 8. This prediction is then subtracted from the actual value of the sample at position x, and the difference is losslessly entropy-coded by either Huffman or arithmetic coding." [7]

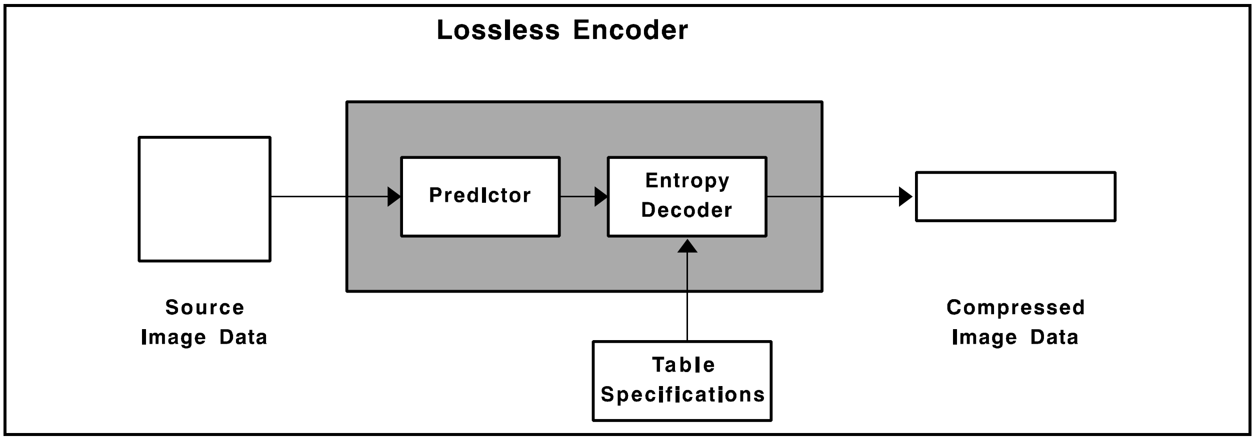

FIGURE 7. Lossless encoder simplified diagram.

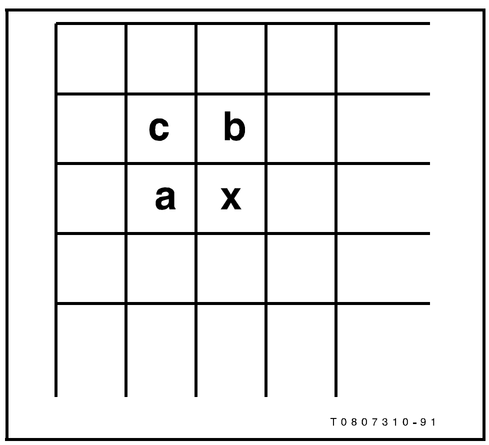

FIGURE 8. 3-sample prediction neighborhood.

4.11 Modes of operation. There are four distinct modes of operation under which the various coding processes are defined:

a. Sequential DCT-based (Type 1, Type 2, Type 3).

[7] Ibid, p. 21.

b. Progressive DCT-based (Type (Effectivity 5)).

c. Lossless (Type (Effectivity 2)).

d. Hierarchical (Type (Effectivity 6)).

The lossless mode of operation was described in 4.10. The other modes of operation are compared in 4.11.1, 4.11.2, and 4.11.3.

4.11.1 Sequential DCT-based mode. For the sequential DCT-based mode, 8x8 sample blocks typically are input block by block from left to right, and block-row by block-row from top to bottom. After a block has been quantized and prepared for entropy coding, all 64 of its quantized DCT coefficients can be entropy encoded immediately and output as part of the compressed image data (as was described in 4.9), minimizing coefficient storage requirements.

4.11.2 Progressive DCT-based mode. (Effectivity 5)

4.11.3 Hierarchical mode. (Effectivity 6)

4.12 Entropy coding alternatives. Two alternative entropy coding procedures are specified:

a. Huffman coding (Type 1, Type 2, Type 3)

b. Arithmetic coding (Type (Effectivity 7))

Huffman coding procedures use Huffman tables, determined by one of the table specifications shown on figures 1 and 3. Arithmetic coding procedures use arithmetic coding conditioning tables, which also may be determined by a table specification. Default values for Huffman tables are specified, but applications may choose tables appropriate for their own environments. Default tables are also defined for the arithmetic coding conditioning.

4.13 Sample precision.

4.13.1 DCT-based processes. For DCT-based processes, two alternative sample precisions are specified:

a. Either 8 bits (Type 1, Type 2)

b. 12 bits (Type 3) per sample

Applications that use samples with other precisions shall use either 8-bit or 12-bit precision by remapping their source image samples. DCT-based implementations that handle 12-bit source image samples are likely to need greater computational resources than those that handle only 8-bit source images.

4.13.2 Lossless processes. "For lossless processes, the sample precision is specified to be from 2 to 16 bits." [8]

4.14 Multiple-component control. Paragraphs 4.9 and 4.10 give an overview of one major subset of the encoding and decoding processes, those that operate on the sample values to achieve compression. Another major subset comprises the procedures that control the order in which the image data from multiple components are processed to create the compressed data and ensure that the proper set of table data is applied to the proper data units in the image. (A data unit is a sample for lossless processes and an 8x8 block of samples for DCT-based processes). [9]

4.14.1 Interleaving multiple components. Figure 9 [10] shows an example of how an encoding process selects between multiple source image components and multiple sets of table data when performing its encoding procedures. The source image in this example consists of the three components A, B, and C, and two sets of table specifications. (This simplified view does not distinguish between the quantization tables and entropy coding tables.)

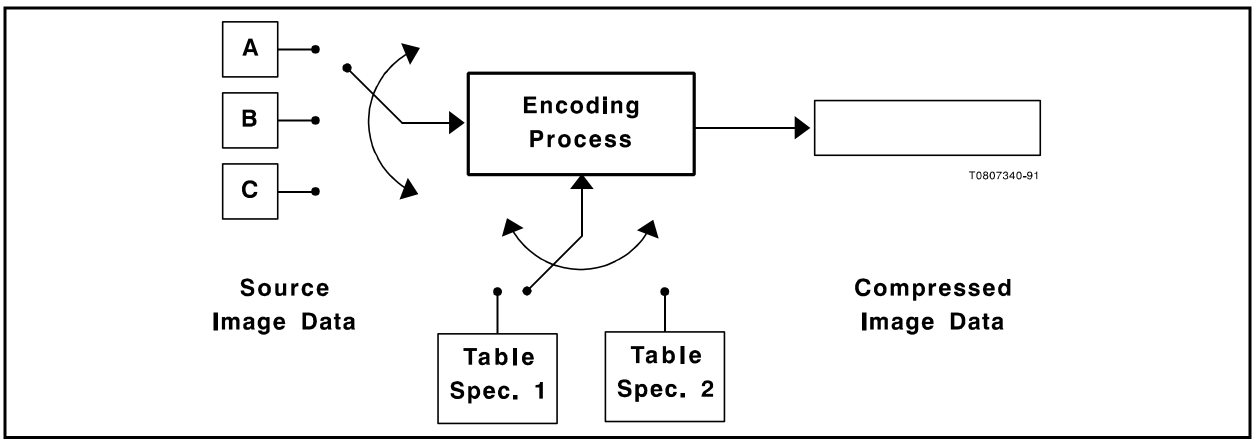

FIGURE 9. Component-interleave and table-switching control.

[8]Ibid, p. 23.

[9]Ibid, p. 24.

[10]Ibid.

In sequential mode, encoding is non-interleaved if the encoder compresses all image data units in component A before beginning component B, and then, in turn, all of B before C. Encoding is interleaved if the encoder compresses a data unit from A, a data unit from B, a data unit from C, then back to A. These alternatives are illustrated on figure 10, which shows a case in which all three image components have identical dimensions: X columns by Y rows, for a total of n data units each. [11]

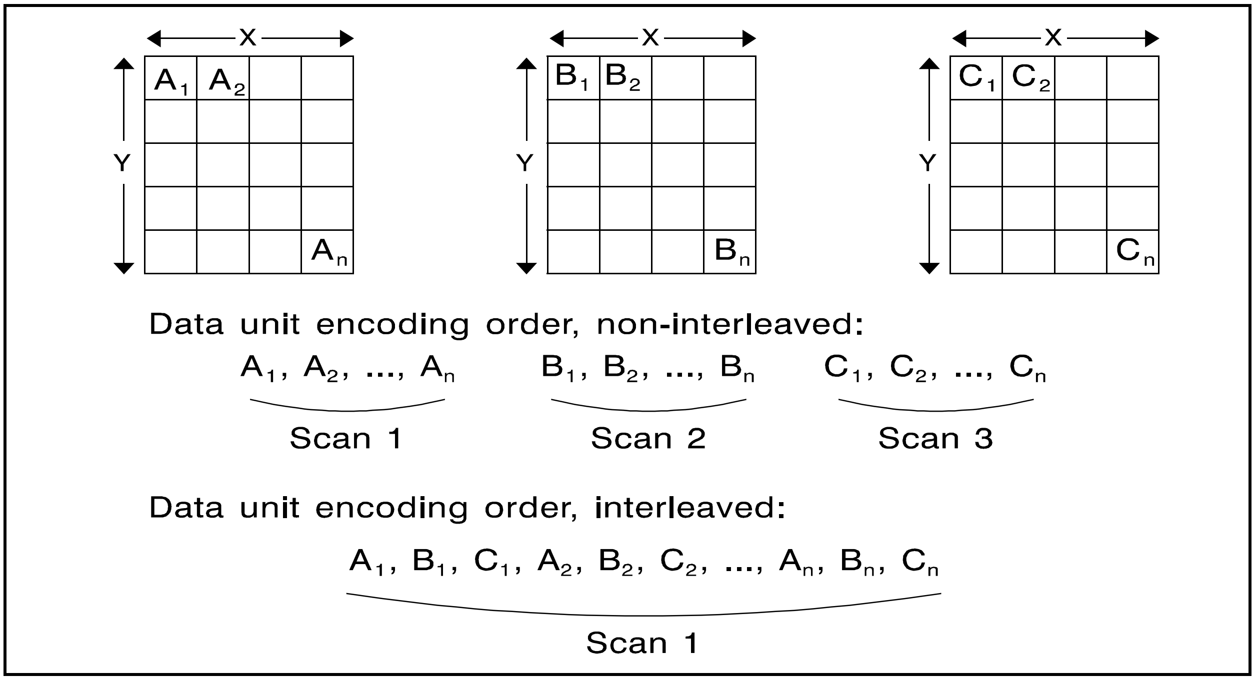

FIGURE 10. Interleaved vs. non-interleaved encoding order.

These control procedures also can handle cases in which the source image components have different dimensions. Figure 11 [12] shows a case in which two of the components, B and C, have half the number of horizontal samples relative to component A. In this case, two data units from A are interleaved with one each from B and C. Cases in which components of an image have more complex sampling relationships, including subsampling in the vertical dimension, can be handled as well.

[11] Ibid, p. 24 and 25.

[12] Ibid, p. 25.

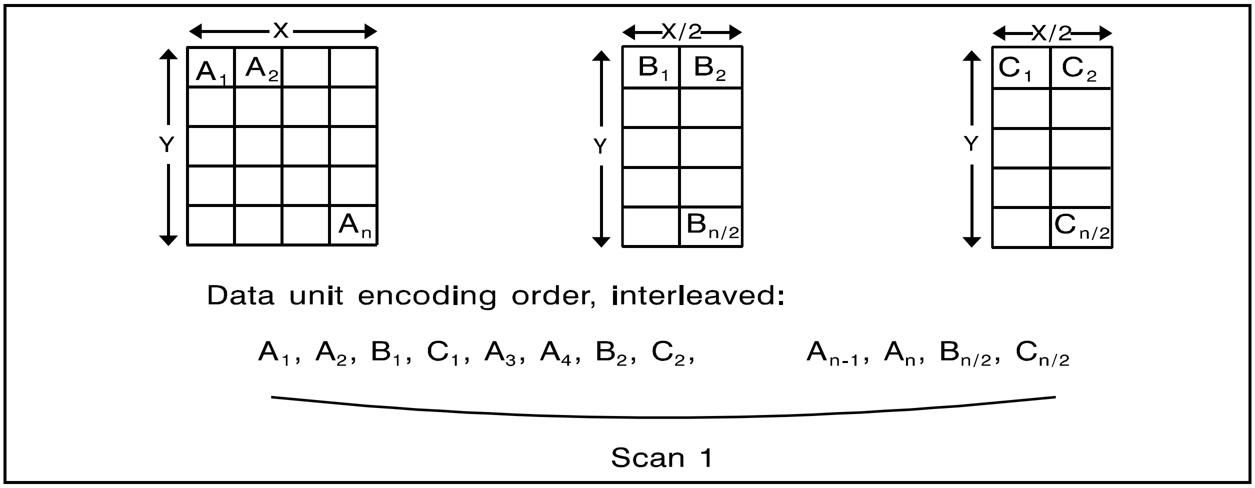

FIGURE 11. Interleaved order for components with different dimensions.

4.14.2 Minimum coded unit. Related to the concepts of multiple-component interleave is the minimum coded unit (MCU). If the compressed image data is non-interleaved, the MCU is defined as one data unit. For example, on figure 10, the MCU for the non-interleaved case is a single data unit. If the compressed data is interleaved, the MCU contains one or more data units from each component. For the interleaved case on figure 10, the first MCU consists of the three interleaved data units, A1, B1, C1. In the example of figure 11, the first MCU consists of the four data units, A1, A2, B1, C1.

4.15 Image, image block, frame, and scan.

4.15.1 Image. An image contains one or more image blocks. Each image block is the same size. For images with more than one image block, it is recommended that block sizes be kept large (512 or more) to minimize the overhead associated with the compressed format. It is also recommended that the block size be chosen to be a multiple of eight times the largest sampling factor, such as 8 or 16, to match the underlying compression process and minimize compression artifacts at the block boundaries.

4.15.2 Image block. Each image block is compressed independently. When the NITF image subheader IMODE field is set to B or P, there is exactly one JPEG stream per image block. When the IMODE field is set to S, there is one JPEG stream per block per band (component).

4.15.3 Frame. A JPEG stream contains only one frame in the cases of sequential and progressive coding processes; a JPEG stream contains multiple frames for the hierarchical mode.

4.15.4 Scan. A frame contains one or more scans. For sequential processes, a scan contains a complete encoding of one or more image components. On figures 10 and 11, the frame consists of three scans when noninterleaved, and one scan if all three components are interleaved together. The frame could also consist of two scans: one with a noninterleaved component, the other with two components interleaved. For progressive processes, a scan contains a partial encoding of all data units from one or more image components. Components shall not be interleaved in progressive mode, except for the DC coefficients in the first scan for each component of a progressive frame. [13]

4.16 Region of interest coding. This standard allows different image blocks within a single image to be compressed using different quantization tables with resulting variation in reconstructed image quality, based on their judged relative importance. In addition, this standard allows different regions within a single image block to be compressed at different rates (Type (Effectivity 3)).

[13] Ibid, p. 27.

5. DETAILED REQUIREMENTS

5.1 Common JPEG encoding and decoding processes. JPEG has two classes of coding processes:

a. DCT-based

b. Lossless

Section 5.1.1 describes requirements common to all DCT-based processes. Section 5.1.2 describes requirements common to all lossless processes. Then specific coding modes are described in 5.2 and subsequent sections.

5.1.1 DCT-based encoding and decoding processes. The following subsections in 5.1.1 describe the requirements for all DCT-based encoding and decoding processes. Specific DCT-based modes are described in subsequent sections:

a. Sequential DCT-based JPEG mode (see 5.2).

b. Progressive DCT-based JPEG mode (see 5.3).

c. Hierarchical JPEG mode (see 5.4).

5.1.1.1 Minimum processing unit. For DCT-based coding modes, the image is processed as a series of 8x8 blocks. For multiple-component imagery, these components may be interleaved. Figure 12 shows a single image component that has been partitioned into blocks for the FDCT computations.

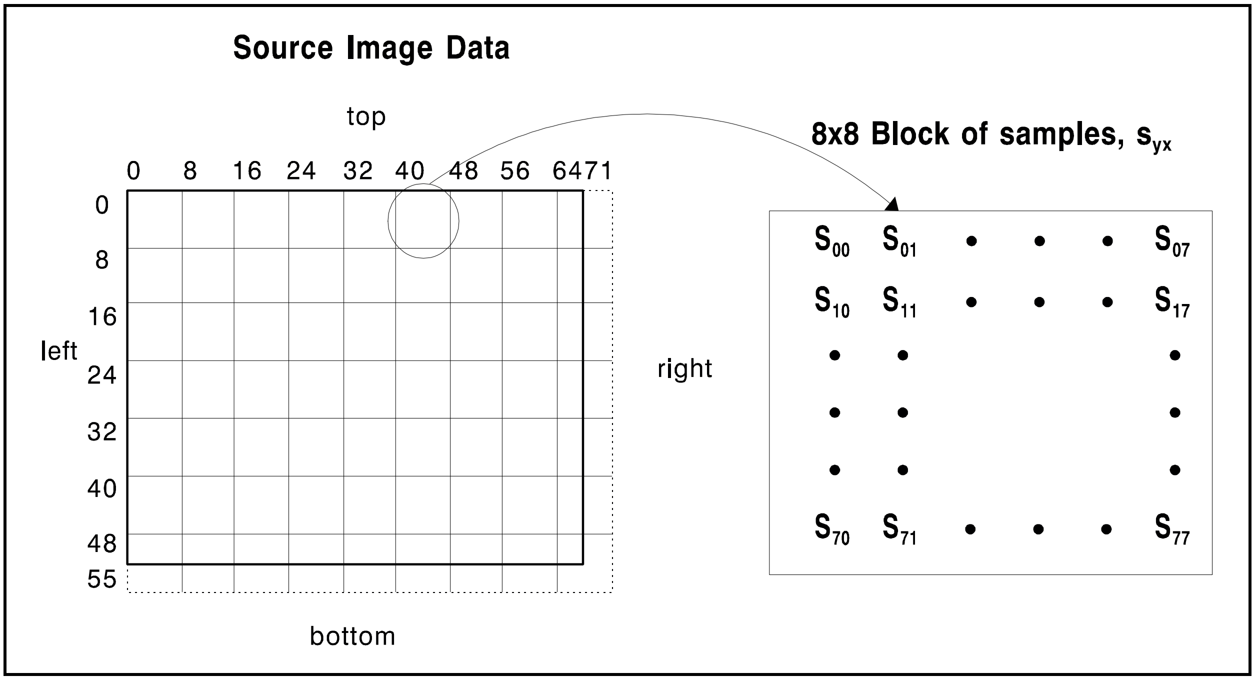

FIGURE 12. Partitioning an example source image with 52 lines and 68 samples into 8x8 blocks.

Figure 12 also defines the orientation of the samples within a block by showing the indices used in the FDCT equation. The definitions of block partitioning and sample orientation also apply to any DCT decoding process and the reconstructed image output. The encoding process shall add samples as necessary at the right and bottom edges of the image to guarantee an integral number of blocks for compression as demonstrated on figure 12. When an image does not contain enough data to fill an 8x8 block along the rightmost or bottom edge, the sample values in the rightmost and/or bottom edges shall be replicated in the extra space to add columns to the right and rows to the bottom. The decoding process shall use header information from the interchange format to identify any such additional samples and remove them. The blocks are processed block by block from left to right, and block-row by block-row from top to bottom.

5.1.1.2 Multiple-component imagery. Multiple-component imagery is any image with more than one component. RGB color and multispectral imagery are important special cases (Effectivity 4).

5.1.1.2.1 RGB color imagery. RGB color imagery can be coded directly or after converting to the YCbCr601 color space. The color space used is recorded in the IREP and IREPBAND fields within the NITF image subheader (defined in MIL-STD-2500).

5.1.1.2.1.1 RGB color space. RGB color imagery can be coded directly as RGB components. Each component shall be compressed at full resolution, with no subsampling, and the components may be interleaved or not. When the components are interleaved, the interleave order is R, G, B with each MCU containing one R 8x8 block, followed by one G block and then one B block. The (i,j) th MCU will contain 3 blocks: R(i,j), G(i,j), B(i,j) where R(i,j) denotes the (i,j) th 8x8 block of the R component.

5.1.1.2.1.2 YCbCr601 color space. RGB color imagery can be coded in the YCbCr601 color space. The following equations specify the ideal functional definition of the forward and inverse transformations. Note that unlike CCIR 601-1, (Y, Cb, Cr) have a full 8-bit dynamic range (0-255) in this standard with no headroom or footroom.

Forward YCbCr601 transformation:

$$
\begin{aligned}
Y &= 0.299R + 0.587G + 0.114B \\
C_b - 128 &= -0.1687R - 0.3313G + 0.500B \\
C_r - 128 &= 0.500R - 0.4187G - 0.0813B
\end{aligned}
$$

The chrominance components can be computed alternatively as color differences:

$$
\begin{aligned}
C_b - 128 &= 0.5643(B - Y) \\
C_r - 128 &= 0.7133(R - Y)
\end{aligned}
$$

Inverse RGB transformation:

$$
\begin{aligned}
R &= Y + 1.402(C_r - 128) \\
G &= Y - 0.34414(C_b - 128) - 0.71414(C_r - 128) \\
B &= Y + 1.772(C_b - 128)
\end{aligned}
$$

These equations contain terms which cannot be represented with perfect accuracy. The accuracy requirements for the combined YCbCr conversion, FDCT, and quantization procedures are specified in JIEO Circular 9008. The accuracy requirements for the combined dequantization, RGB conversion, and IDCT also are specified in JIEO Circular 9008.

5.1.1.2.1.3 Interleave order for YCbCr. The chrominance components (Cb, Cr) may be subsampled by 2 horizontally, vertically, or both relative to the luminance component (Y). The components may be interleaved or not. When the components are interleaved, the interleave order is Y, Cb, Cr with each MCU containing one or more Y 8x8 blocks, followed by one Cb block and then one Cr block. Depending on the subsampling, each MCU will contain three, four, or six blocks.

The (i,j)th MCU will contain:

No subsampling:

$$Y(i,j),\ Cb(i,j),\ Cr(i,j)$$

Horizontal subsampling:

$$Y(i,2j),\ Y(i,2j+1),\ Cb(i,j),\ Cr(i,j)$$

Vertical subsampling:

$$Y(2i,j),\ Y(2i+1,j),\ Cb(i,j),\ Cr(i,j)$$

Both horizontal and vertical subsampling:

$$Y(2i,2j),\ Y(2i,2j+1),\ Y(2i+1,2j),\ Y(2i+1,2j+1),\ Cb(i,j),\ Cr(i,j)$$

where Y(i,j) denotes the (i,j)th 8x8 block of the Y component.

5.1.1.2.1.4 Downsampling filter. The downsampled components are generated as the output of the downsampling filter, illustrated on figure 13. The associated division indicates truncation, not rounding.

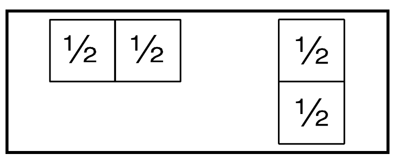

FIGURE 13. Low pass filter for downsampling.

When downsampling horizontally, the left sample on figure 13 should be aligned with the left column of the higher resolution image. When downsampling vertically, the top sample on figure 13 should be aligned with the top row of the higher resolution image. The filter is then applied to each pair of samples along the row, or column, and the filter output generates the lower resolution image. If the image being downsampled has an odd width or length, the odd dimension is increased by one by sample replication on the right edge or bottom row before downsampling. If both horizontal and vertical reductions are required, they are done in sequence - first horizontal and then vertical. Because of the filter structure, the downsampled chrominance samples are centered between the luminance samples.

5.1.1.2.1.5 Upsampling filter. The upsampling filter increases the spatial resolution by a factor of two horizontally, vertically or both. Sample replication is used for the upsampling filter, as illustrated on figure 14.

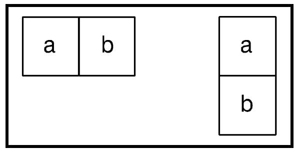

FIGURE 14. Sample positions for upsampling rules.

The rule for calculating the upsampled value is Pa = Pb = Ra where Pa is the upsampled value at location a, Pb is the upsampled value at location b, and where Ra is the sample value of the lower resolution image. The top left sample of the upsampled image matches the top left sample of the lower resolution image. For each sample in the lower resolution image, two adjacent samples are output along the row, or column. The upsampling procedure always doubles the line length or the number or rows. If both horizontal and vertical expansions are required, they are done in sequence. Note that either order produces identical results.

5.1.1.2.2 Multispectral imagery. (Effectivity 4)

5.1.1.3 FDCT and IDCT.

5.1.1.3.1 Level shift. Before an encoding process computes the FDCT for a block of unsigned source image samples, the sample shall be level shifted to a signed representation by subtracting 2^{P-1}, where P is the input sample precision (eight or 12). Thus, when P = 8, the level shift is by 128 and when P = 12, the level shift is by 2048. After the level shift, the expected DC coefficient value is zero for uniformly distributed sample values, rather than mid-range. After a decoding process computes the IDCT and produces a block of reconstructed image samples, an inverse level shift shall restore the samples to their original unsigned representation.[14]

5.1.1.3.2 FDCT and IDCT transformations. The following equations specify the ideal functional definition of the FDCT and the IDCT.

FDCT:

$$S_{vu} = \frac{1}{4} C_u C_v \sum_{x=0}^{7} \sum_{y=0}^{7} s_{yx} \cos\frac{(2x+1)u\pi}{16} \cos\frac{(2y+1)v\pi}{16}; \quad u,v = 0,\ldots,7$$

IDCT:

$$s_{yx} = \frac{1}{4} \sum_{u=0}^{7} \sum_{v=0}^{7} C_u C_v S_{vu} \cos\frac{(2x+1)u\pi}{16} \cos\frac{(2y+1)v\pi}{16}; \quad x,y = 0,\ldots,7$$

[14] Ibid, p. 49.

where

$$C_u, C_v = \frac{1}{\sqrt{2}} \ \ for\ \ u,v=0; \quad C_u, C_v = 1 \ \ otherwise.$$

These equations contain terms that cannot be represented with perfect accuracy. The accuracy requirements for the combined FDCT and quantization procedures are specified in JIEO Circular 9008. The accuracy requirements for the combined dequantization and IDCT also are specified in JIEO Circular 9008.

5.1.1.4 DCT coefficient quantization and dequantization. After the FDCT is computed for a given block, each of the 64 resulting DCT coefficients, S_{vu} is quantized by an independent uniform quantizer. The quantizer step size for each coefficient is the value of the corresponding element, Q_{vu}, from the quantization table.[15]

Quantization:

$$Sq_{vu} = round\left(\frac{S_{vu}}{Q_{vu}}\right)$$

Dequantization:

$$\boldsymbol{R_{vu} = Q_{vu} \times Sq_{vu}}$$

5.1.1.5 DC coefficient - differential encoding. The DC coefficient, S_{00}, is processed differently than the 63 AC coefficients. The quantized DC coefficient from the previously encoded block (of the same component), PRED, is used to predict the current quantized DC coefficient, Sq_{00}, and this difference, DIFF, is encoded.

$$DIFF = Sq_{00} - PRED$$

where $PRED = Sq_{00}$ from the previous block (of the same component).

At the beginning of the image and at the beginning of each restart interval, PRED is initialized to zero.

[15] Ibid, p. 50.

5.1.1.6 AC coefficients - zig-zag scan order. After quantization and in preparation for entropy encoding, the quantized AC coefficients are scanned in a zig-zag order to maximize the run lengths of zero quantized coefficients. The zig-zag order AC coefficients are denoted ZZ_k; $k = 1, \ldots, 63$ and the zig-zag sequence is specified as follows:

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

so that:

$$
\begin{aligned}
ZZ_1 &= Sq_{01} \\
ZZ_2 &= Sq_{10} \\
ZZ_3 &= Sq_{20} \\
ZZ_4 &= Sq_{11} \\
&\;\;\vdots \\
ZZ_{63} &= Sq_{77}
\end{aligned}
$$

5.1.2 Lossless encoding and decoding processes. (Effectivity 2)

5.2 Sequential DCT-based JPEG mode.

5.2.1 Control procedures for the sequential DCT-based mode. The control procedures for encoding and decoding an image and its constituent parts, the frame, scan, restart interval, and MCU, are given on figures 15 and 16. The procedure for encoding (decoding) a MCU repetitively calls the procedure for encoding (decoding) a data unit. For DCT-based modes, the data unit is an 8x8 block of samples.

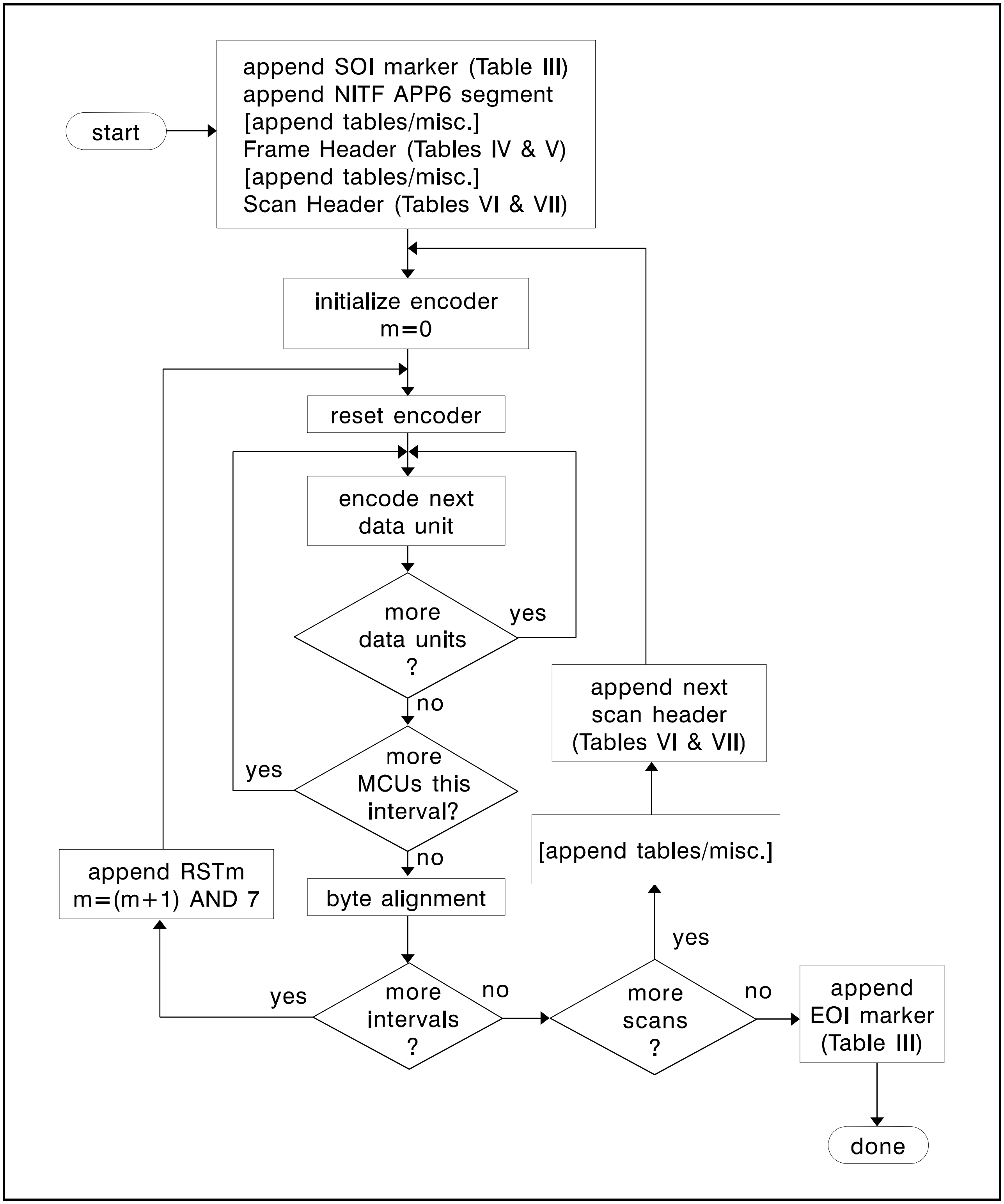

FIGURE 15. Simplified encode flow chart.

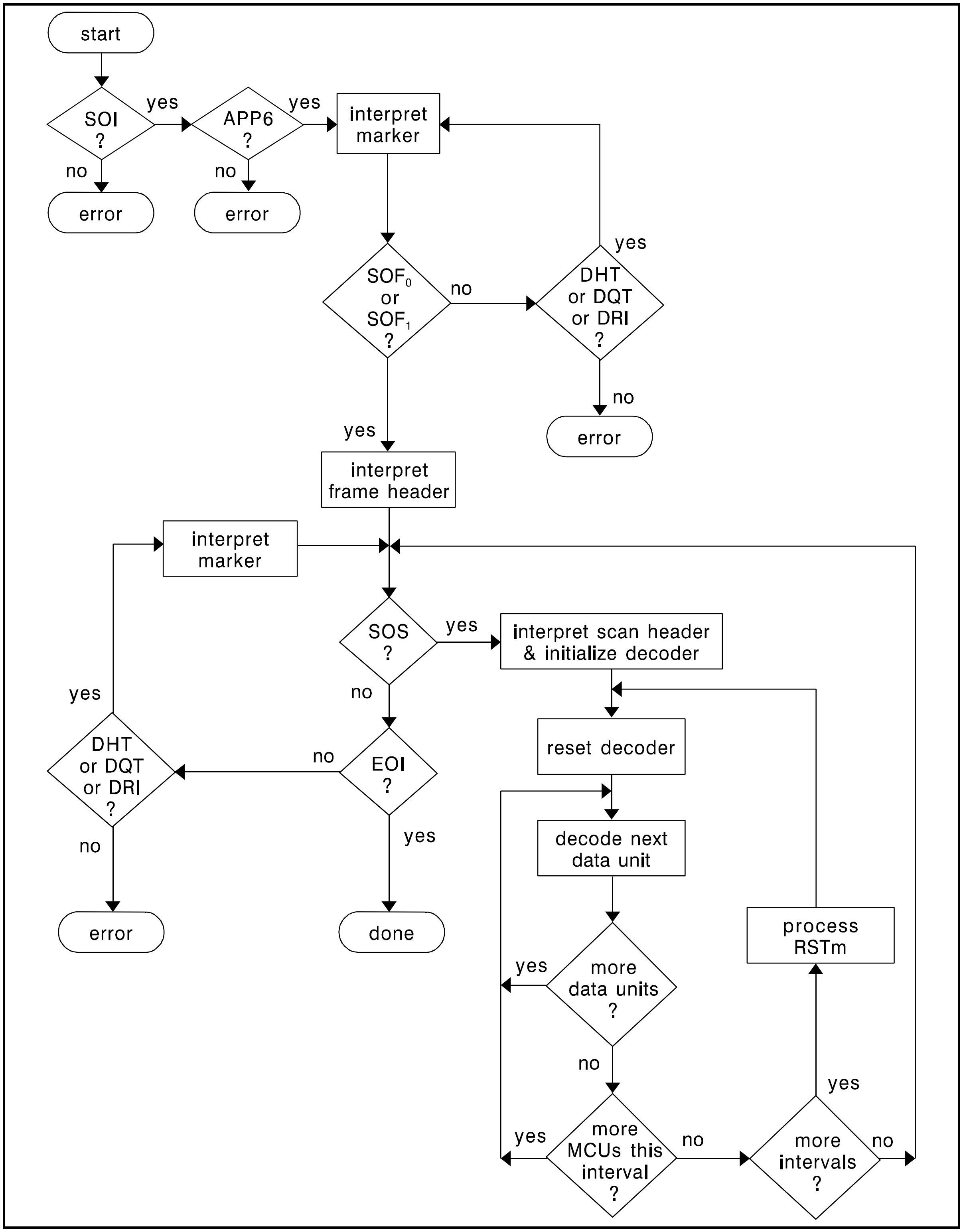

FIGURE 16. Simplified decode flow chart.

5.2.2 Procedure for encoding and decoding an 8x8 block data unit. The steps in the sequential DCT-based encoding process for an 8x8 block data unit are: [16]

a. Calculate FDCT and quantize.

b. Encode DC coefficient for 8x8 block.

c. Encode AC coefficients for 8x8 block.

The steps in the decoding process are:

a. Decode DC coefficient for 8x8 block.

b. Decode AC coefficients for 8x8 block.

c. Inverse quantize and calculate IDCT.

5.2.2.1 FDCT. The mathematical definition of the FDCT is given in 5.1.1.3.2. Before computing the FDCT, a level shift is performed by subtracting 2^{P-1}, where P is the sample precision (eight or 12). For 8-bit input precision, the level shift is by 128, and for 12-bit input precision, the level shift is by 2048. [17]

5.2.2.2 IDCT. The mathematical definition of the IDCT is given in 5.1.1.3.2. After computing the IDCT, a level shift is performed by adding 2^{P-1}, where P is the sample precision (eight or 12). For eight-bit output precision, the level shift is by 128, and for 12-bit output precision, the level shift is by 2048. [18]

5.2.2.3 Quantization rules. The DCT coefficients are uniformly quantized.

5.2.2.3.1 Quantization. Quantization is accomplished by dividing each DCT coefficient value by the quantization table value for that coefficient and rounding the result. The quantized DCT coefficient values are signed, two's complement integers with 11-bit precision for 8-bit input precision and 15-bit precision for 12-bit input precision.

5.2.2.3.2 Dequantization. Dequantization is accomplished by multiplying each quantized coefficient value by the quantization table value for that coefficient.

5.2.2.4 Quantization tables. Default or custom tables can be used.

---

[16] Ibid, p. 113.

[17] Ibid, p. 114.

[18] Ibid, p. 49 and p. 134.

5.2.2.4.1 Default quantization tables. Default quantization tables, suitable for most applications, are provided in appendix A. When the default tables are used, they are not placed in the compressed data. The use of the default table(s) is recorded in the COMRAT field within the NITF image subheader (defined in MIL-STD-2500).

5.2.2.4.2 Custom quantization tables. For those applications when the default tables are not appropriate, custom quantization tables can be used. The tables then are placed in the compressed data according to the format specified in this standard.

5.2.2.5 Entropy encoder/decoder. Two distinct entropy coders are possible. Paragraph 5.2.2.5.1 specifies Huffman coding while 5.2.2.5.2 specifies arithmetic coding (Effectivity 7). Huffman coding is required at this time and arithmetic coding is anticipated as a future requirement.

5.2.2.5.1 Huffman coding.

5.2.2.5.1.1 Coding models for Huffman coding. After quantization, the DCT coefficients are prepared for Huffman coding. The coefficients are mapped into symbols as described below and these symbols then are Huffman coded. The encoded symbol then usually is followed by a variable length bit field, which, together with the symbol, fully determines the coefficient(s) being encoded. The decoder first decodes the symbol following the Huffman decoding procedure. Based on the value of the symbol, any additional bit field is interpreted to reconstruct the quantized DCT coefficient(s).

5.2.2.5.1.2 Forming the DC symbol. Instead of assigning individual Huffman codes to each DIFF value, the DIFF values are categorized based on magnitude ranges and this category, CAT, is Huffman encoded. Additional bits are appended to specify the actual DIFF value within the category. For 8-bit sample precision, the DIFF values fall within the range (-2047, 2047) and are grouped into 12 categories, 0-11, as shown in table I. When the sample precision is 12 bits, the DIFF values have a larger dynamic range (-32767, 32767) and are grouped into 16 categories, 0-15, also shown in table I.

TABLE I. Categories.

Category, CAT	Values
0	0
1	-1, 1
2	-3, -2, 2, 3
3	-7…-4, 4…7,
4	-15…-8, 8…15
5	-31…-16, 16…31
6	-63…-32, 32…63
7	-127…-64, 64…127
8	-255…-128, 128…255
9	-511…-256, 256…511
10	-1023…-512, 512…1023
11	-2047…-1024, 1024…2047
12	-4095…-2048, 2048…4095
13	-8191…-4096, 4096…8191
14	-16383…-8192, 8192…16383
15	-32767…-16384, 16384…32767

5.2.2.5.1.3 Encoding the DIFF value. For the DC coefficient, the symbol used for Huffman encoding is CAT. For any given DIFF value, an additional bit sequence is appended to the Huffman codeword to identify uniquely which difference in that category actually occurred. The number of extra bits is given by CAT itself, with zero being an allowed value. The extra bits are appended to the least significant bit (LSB) of the Huffman code, most significant bit (MSB) first. When DIFF is positive, the

CAT low order bits of DIFF are appended. When DIFF is negative, the CAT low order bits of (DIFF-1) are appended. Note that the most significant bit of the appended bit sequence is zero for negative differences and one for positive differences.

EXAMPLE.

When DIFF = 2 = 00000…010, CAT = 2 and the CAT low order bits of DIFF are 10.

When DIFF = -2, CAT = 2, DIFF-1 = -3 = 11111…101 and the CAT low order bits of DIFF-1 are 01.

5.2.2.5.1.4 Forming the AC symbol. Each non-zero AC coefficient in the vector of zig-zag ordered coefficients is described by a composite 8-bit symbol, V, of the form:

$$V = (RUN, CAT) = 16 \times RUN + CAT$$

as described in table II.

TABLE II. AC symbols.

<table>
<tr><th></th><th></th><th colspan="15">CAT</th></tr>
<tr><th></th><th></th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>10</th><th>11</th><th>12</th><th>13</th><th>14</th></tr>
<tr><td></td><td>0</td><td>EOB</td><td colspan="10" rowspan="16">V = 16 x RUN + CAT</td><td colspan="4" rowspan="16">additional symbols for 12-bit sample data</td></tr>
<tr><td></td><td>1</td><td>XXX</td></tr>
<tr><td></td><td>2</td><td>XXX</td></tr>
<tr><td></td><td>3</td><td>XXX</td></tr>
<tr><td></td><td>4</td><td>XXX</td></tr>
<tr><td></td><td>5</td><td>XXX</td></tr>
<tr><td>R</td><td>6</td><td>XXX</td></tr>
<tr><td>U</td><td>7</td><td>XXX</td></tr>
<tr><td>N</td><td>8</td><td>XXX</td></tr>
<tr><td></td><td>9</td><td>XXX</td></tr>
<tr><td></td><td>10</td><td>XXX</td></tr>
<tr><td></td><td>11</td><td>XXX</td></tr>
<tr><td></td><td>12</td><td>XXX</td></tr>
<tr><td></td><td>13</td><td>XXX</td></tr>
<tr><td></td><td>14</td><td>XXX</td></tr>
<tr><td></td><td>15</td><td>ZRL</td></tr>
</table>

The 4 least significant bits define a category, CAT, for the amplitude of this nonzero coefficient. For 8-bit image data, the AC coefficients can be shown to fall within the range (-1023, 1023) and are grouped into ten categories, 1-10, using table I. For 12-bit image data, the AC coefficients have a larger dynamic range (-16383, 16383) and are grouped into the 14 categories 1-14, also shown in table I. CAT = 0 is not valid here because we only are considering the next nonzero coefficient. The 4 most significant bits define RUN, the run-length of zero coefficients between nonzero coefficients, which gives the position of this coefficient relative to the previous nonzero coefficient (in this block). If the run length of zero coefficients exceeds 15, then multiple symbols are used. The special zero run length (ZRL) symbol (15, 0) = 240 is defined to represent a run length of 15 zero coefficients followed by a coefficient of zero amplitude that can be interpreted as a run length of 16 zero coefficients. For example a (RUN = 35, CAT = 5) pair would result in three symbols:

a. (15,0)

b. (15,0)

c. (3,5)

In addition, a special value (0,0) = 0 is used to code the end-of-block (EOB), signaling that all remaining coefficients in the block are zero. Note that if ZZ_{63}, the last AC coefficient in the block, is nonzero, then no EOB symbol is encoded.

5.2.2.5.1.5 Encoding the AC coefficient values. For any given nonzero AC coefficient value, an additional bit field is appended to the Huffman codeword to identify uniquely which coefficient value within that category actually occurred. The procedure is the same as for the DIFF value (see 5.2.2.5.1.3). The number of extra bits is given by CAT itself, and the bits are appended to the LSB of the Huffman code, MSB first. When ZZ_k is positive, the CAT low order bits of ZZ_k are appended. When ZZ_k is negative, the CAT bits of (ZZ_k - 1). Note that the MSB of the appended bit sequences is zero for negative coefficients and one for positive coefficients.

5.2.2.5.1.6 Huffman codes. The DC and AC coefficients are represented, in part, by symbols that are Huffman coded. Huffman codes are the most efficient codes possessing the prefix property. That is, given a set of symbols and their associated probability of occurrence, a set of Huffman codes can be generated to minimize the average number of bits required to represent a typical ensemble of symbols, so that each code will not be a prefix to any other valid code. This allows a bit stream to be parsed, a bit at a time, to decode the encoded symbols. This algorithm never has more than 256 symbols, and the code lengths are limited to 16 by design.

5.2.2.5.1.7 Huffman table generation. The Huffman codes are generated from two tables, BITS and HUFFVAL. BITS is a list of 16 8-bit values defining the number of Huffman codes of each size, one through 16. HUFFVAL is a list of 8-bit symbols; one symbol for each code in increasing code length order. A method of generation is specified so that, given BITS and HUFFVAL, the associated codes are defined uniquely.

5.2.2.5.1.8 Default BITS and HUFFVAL tables. Default values, suitable for most applications are provided in appendix B. When the default tables are used, they are not placed in the compressed data.

5.2.2.5.1.9 Custom BITS and HUFFVAL tables. For those applications when the default tables are not appropriate, custom Huffman tables can be used. In that case, the procedure described in appendix C must be followed to generate the custom BITS and HUFFVAL tables which are then placed in the compressed data according to the format specified in this standard.

5.2.2.5.1.10 Building the Huffman coding tables. The Huffman encoding tables, EHUFCO and EHUFSI, are built from BITS and HUFFVAL following the procedure in appendix D.

5.2.2.5.1.11 Huffman encoding. The Huffman code for V is the EHUFSI(V) rightmost bits of EHUFCO(V). The MSB of the Huffman code is placed towards the MSB of the byte in the bit stream, 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. Additional bit sequences associated with Huffman codes are appended with the MSB adjacent to the LSB of the Huffman code.

5.2.2.5.1.12 Huffman decoding. A Huffman decoder may be implemented in many ways, and one possibility is described as an example. This decoding method uses three tables: MINCODE, MAXCODE, and VALPTR, which are generated from BITS and HUFFCODE as shown on figure 17 [19]. HUFFCODE is built by a procedure defined in appendix D.

[19]Ibid, p. 137.

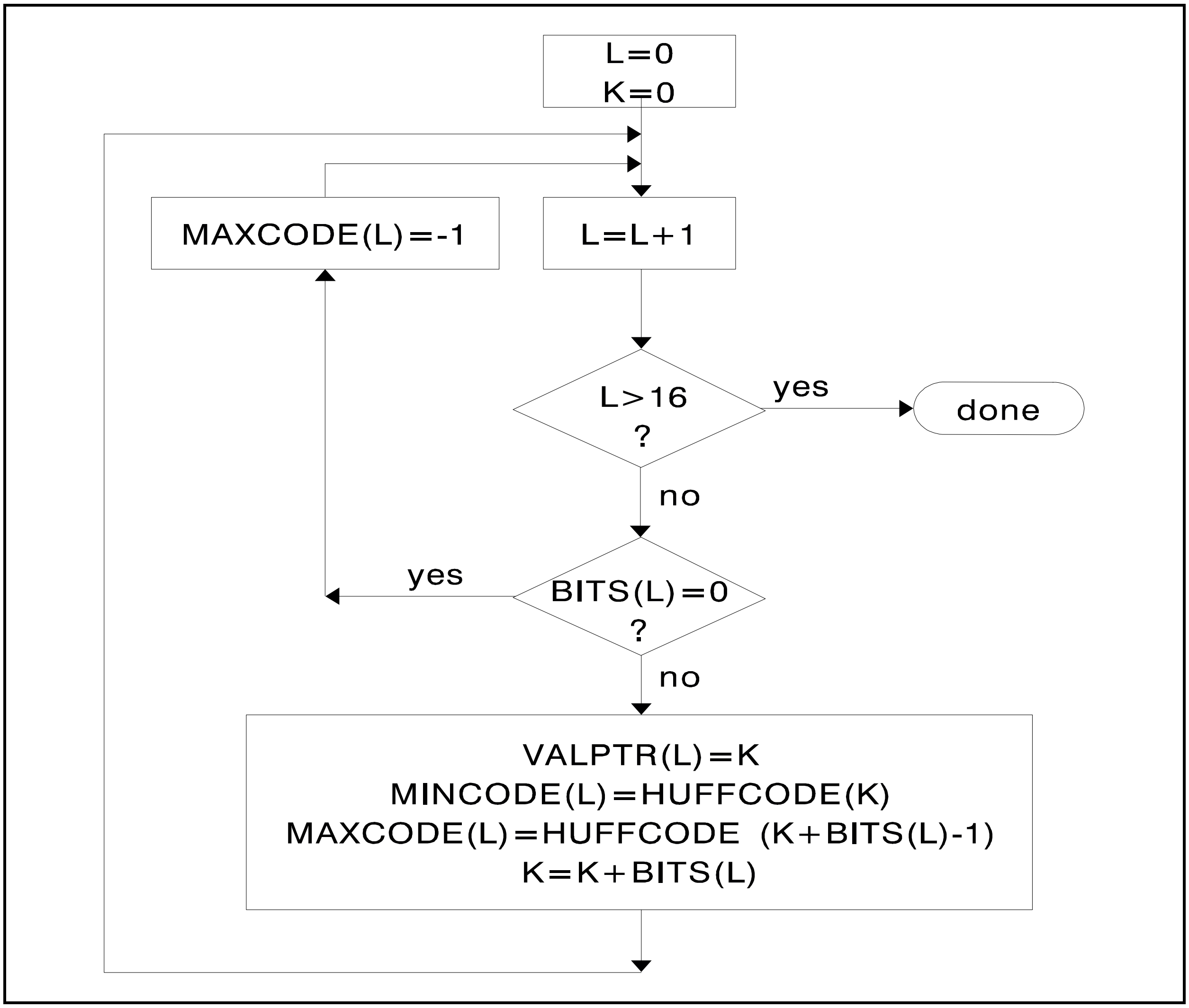

FIGURE 17. Decoder table generation.

To decode the next symbol (value) the compressed data string is examined one bit at a time. First it is determined if a one-bit code is valid, and, if not, the next bit is fetched from the data string and the resulting two-bit code is examined. This procedure is repeated until a valid code is found. The procedure is shown on figure 18 [20], where the procedure NEXTBIT returns the next code bit from the bit stream. The notation (SL CODE 1) on figure 18 indicates a shift left of CODE by one bit in position.

[20]Ibid, p. 138.

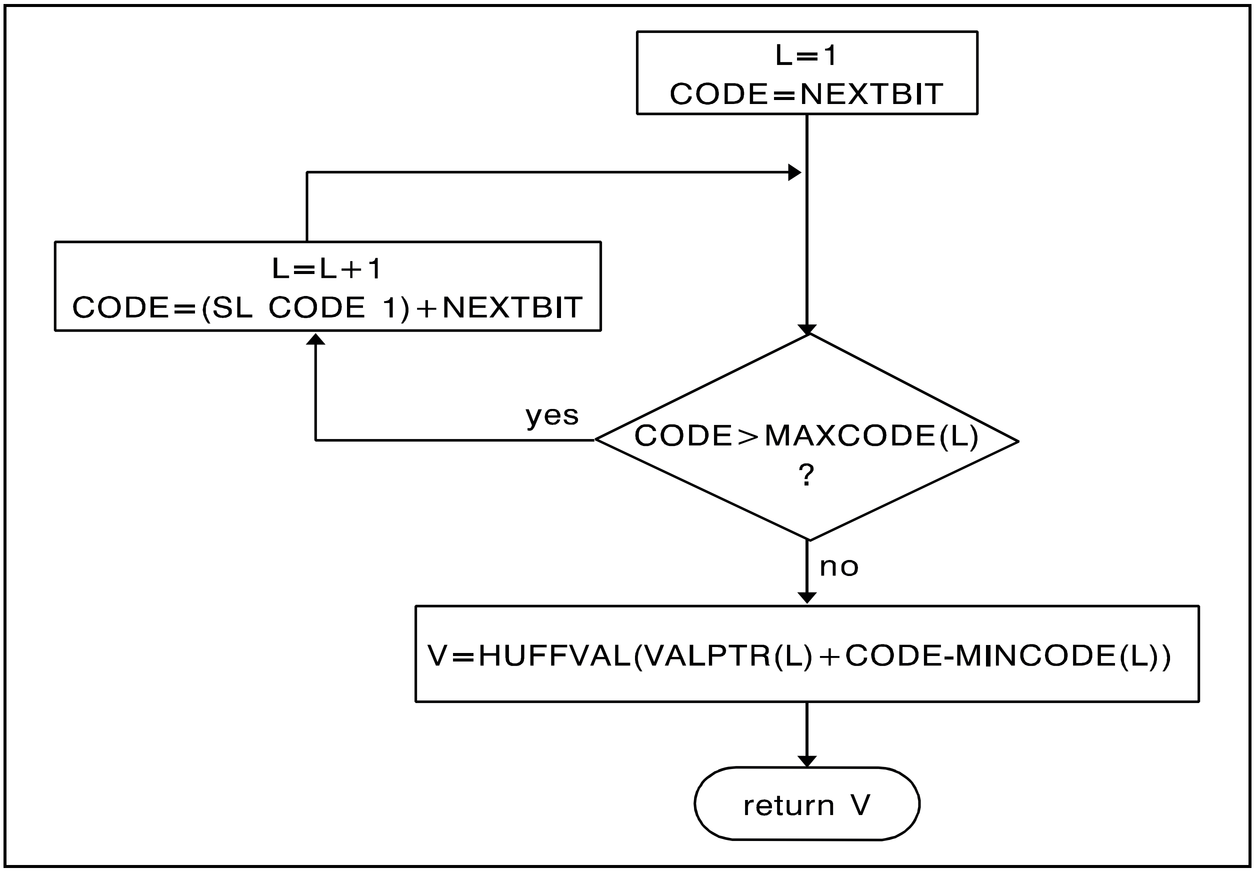

FIGURE 18. Huffman decode procedure.

5.2.2.5.2 Arithmetic coding. (Effectivity 7)

5.2.3 Compressed data interchange format. Section 5.2.2 has described how to code and decode a single 8x8 block. To encode a complete image, an interchange format has been defined that includes additional information, conveyed via marker codes, together with the entropy-coded data. The interchange format consists of an ordered collection of markers, parameters, and entropy-coded segments.

5.2.3.1 Marker codes. Marker codes always are byte aligned and are preceded by the 0xFF marker prefix byte. Any marker optionally may be preceded by any number of fill bytes, which are bytes assigned code 0xFF. All markers are assigned two-byte codes, a 0xFF byte followed by a second byte that is not equal to zero or 0xFF. The marker codes fall into two classes: codes without fields and codes followed by a variable length parameter segment. Table III shows the marker codes that are valid for the sequential DCT-based encoding process.

TABLE III. Marker codes for sequential DCT-based mode.

0xFFC0	SOF_0 - Baseline DCT
0xFFC1	SOF_1 - Extended sequential DCT
0xFFC4	DHT - Define Huffman Table(s)
0xFFD0	RST_0 - Restart with modulo 8 count 0
0xFFD1	RST_1 - Restart with modulo 8 count 1
0xFFD2	RST_2 - Restart with modulo 8 count 2
0xFFD3	RST_3 - Restart with modulo 8 count 3
0xFFD4	RST_4 - Restart with modulo 8 count 4
0xFFD5	RST_5 - Restart with modulo 8 count 5
0xFFD6	RST_6 - Restart with modulo 8 count 6
0xFFD7	RST_7 - Restart with modulo 8 count 7
0xFFD8	SOI - Start of Image
0xFFD9	EOI - End of Image
0xFFDA	SOS - Start of Scan
0xFFDB	DQT - Define Quantization Table(s)
0xFFDD	DRI - Define Restart Interval
0xFFE6	APP_6 - NITF application segment
0xFFFE	COM - Comment

5.2.3.2 Byte stuffing. In the compressed data, any non-zero value following one or more 0xFF bytes is defined as a marker code, where 0x indicates a hexadecimal number. Therefore, whenever, in the course of normal encoding, the byte value 0xFF is created in the code string, a 0x00 byte is stuffed into the code string, after the created 0xFF, to prevent the false detection of a marker code. If an 0x00 byte is detected after an 0xFF byte, the decoder must discard it. If the byte is not zero, a marker code has been detected and shall be interpreted to the degree needed to decode the data.

5.2.3.3 Format of a JPEG compressed image within an NITF file. The format for NITF image data compressed with the sequential DCT-based JPEG mode differs based on the number of blocks, bands, and IMODE value (B,P,S). These different cases are described below.

5.2.3.3.1 Single block JPEG compressed format. The format for NITF single block image data compressed with the sequential DCT-based JPEG mode is shown on figure 19.

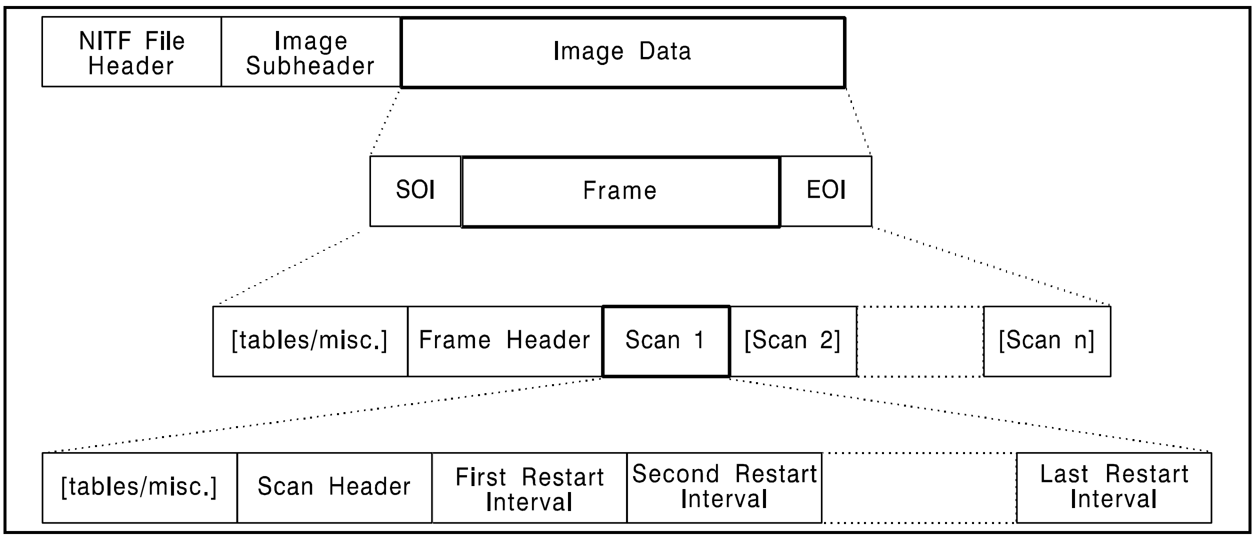

FIGURE 19. NITF single block file structure (IMODE = B or P).

5.2.3.3.1.1 Single block image data format. The top level of figure 19 specifies that the JPEG compressed data is contained in the Image Data Field of the NITF file. The second level of figure 19 specifies that the single block image format shall begin with an SOI marker, shall contain one frame, and shall end with an EOI marker.

5.2.3.3.1.2 Frame format. The third level of figure 19 specifies that a frame shall begin with a frame header and shall contain one or more scans. A frame header may be preceded by one or more table-specification or miscellaneous marker segments. NITF does not allow the use of the JPEG DNL segment which, when present, would follow the first scan in the frame.

5.2.3.3.1.3 Scan format. The fourth level of figure 19 specifies that a scan shall begin with a scan header and shall contain one or more restart intervals. A scan header may be preceded by one or more table-specification or miscellaneous marker segments. When the NITF image subheader IMODE field is set to B, there shall be n scans within the frame, one for each of the components (n = 1 or 3). When the IMODE field is set to P, there shall be a single scan within the frame consisting of three interleaved components.

5.2.3.3.1.4 Restart intervals. Following the scan header, each scan shall be encoded as a series of one or more restart intervals. A restart interval is a self-contained entropy-coded data segment that can be decoded independently from the other intervals. Restart intervals are used for error recovery (6.3). If the image were encoded as a single interval, any transmission error would render all subsequent image data unusable. When several restart intervals are used, the effects of an error can be contained within a single interval. The restart interval is defined by the DRI marker in a miscellaneous marker segment, and each interval, except the last, shall be followed by a marker code (RST_m, $m = 0,\ldots,7$) where m is the interval count modulo eight. In JPEG, restart intervals are optional but NITF requires the use of restart marker codes with a restart interval no larger than the number of MCUs per block-row.

5.2.3.3.1.5 Byte alignment. To achieve byte alignment at the end of a restart interval, any incomplete byte is padded with one-bits. If this padding creates a 0xFF value, a zero byte is stuffed (see 5.2.2.2) before adding the following 0xFF prefix and marker code to prevent a decoder from interpreting this incomplete byte as a marker code.

5.2.3.3.2 Multiple block JPEG compressed format. The format for NITF multiple block image data compressed with the sequential DCT-based JPEG mode is shown on figure 20 for IMODE = B or P. The corresponding format when IMODE = S is shown on figure 21. Default quantization tables and Huffman tables shall apply to all blocks unless the compressed stream for that block includes custom table definitions. There shall be no carryover of custom tables between blocks so that custom tables must be included in each block where the defaults are not used.

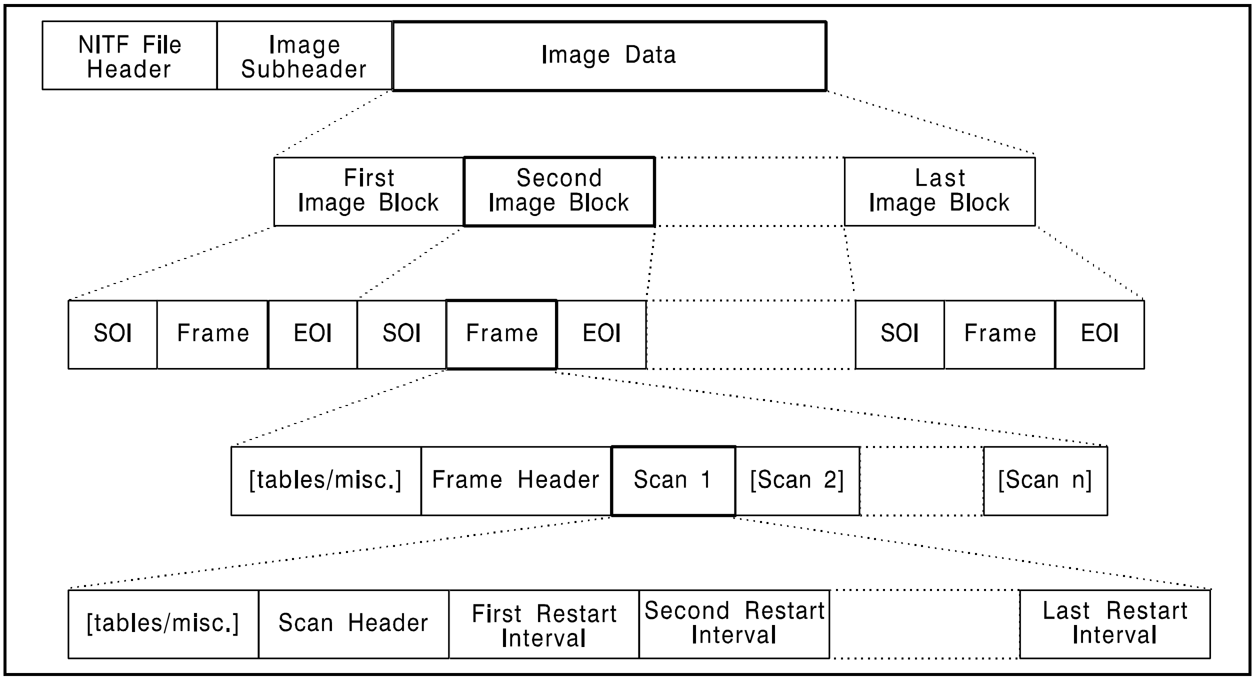

FIGURE 20. NITF multiple block file structure (IMODE = B or P).

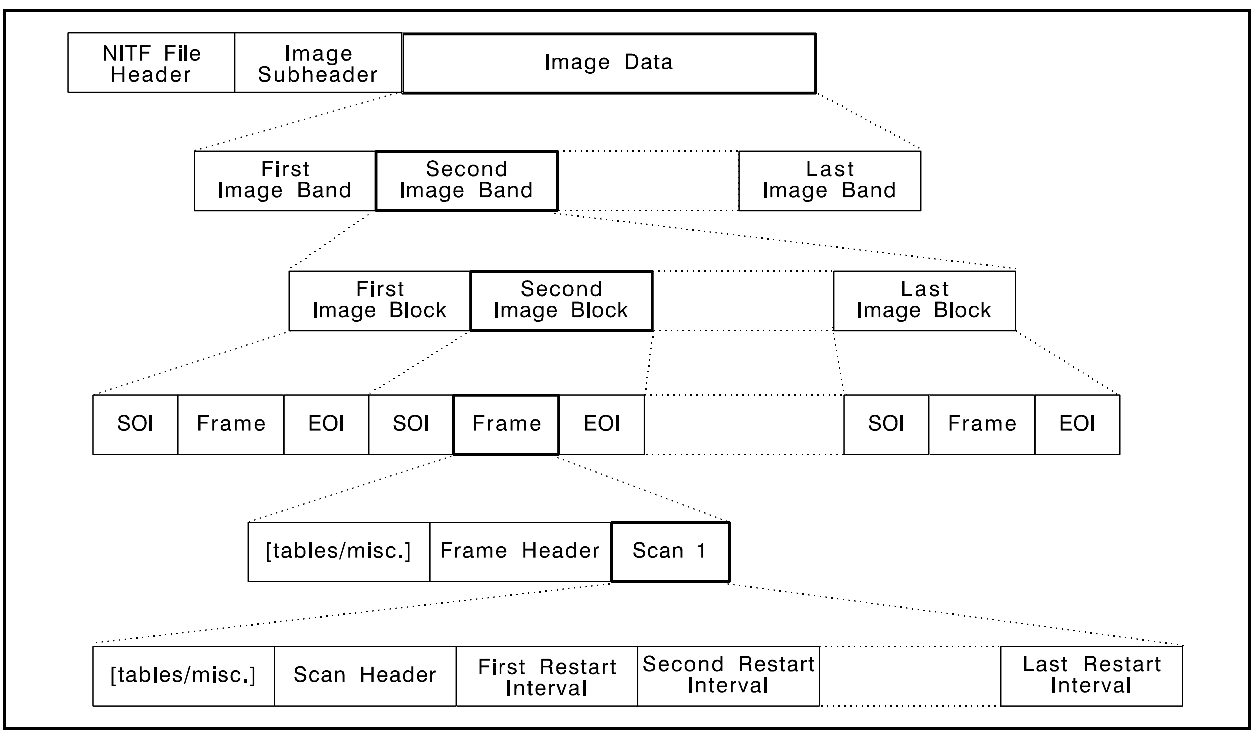

FIGURE 21. NITF multiple block file structure (IMODE = S).

5.2.3.3.2.1 Multiple block image data format (IMODE = B or P). The top level of figure 20 specifies that the JPEG compressed data is contained in the Image Data Field of the NITF file. The second level of figure 20 specifies that this multiple block image format shall begin with the compressed data for the first image block and shall be followed by the compressed data for each image block, one after the other, left to right, top to bottom. The third level of figure 20 specifies that each compressed block shall begin with an SOI marker, shall contain one frame, and shall end with an EOI marker. The format below this level is identical to the single block case previously described in 5.2.3.3.1.

5.2.3.3.2.2 Multiple block image data format (IMODE = S). The use of this IMODE requires that the image contain multiple blocks and multiple bands, otherwise IMODE shall be set to B or P. The top level of figure 21 specifies that the JPEG compressed data is contained in the Image Data Field of the NITF file. The second level of figure 21 specifies that this multiple block image format shall begin with the compressed data for the first image band and shall be followed by the compressed data foe each image band, one after the other, first to last. The third level of figure 21 specifies that each compressed image band shall consist of the compressed data (for that band) for each image block, one after the other, left to right, top to bottom. The fourth level of figure 21 specifies that each compressed block shall begin with an SOI marker, shall contain one frame, and shall end with an EOI marker. The format below this level is identical to the single block case previously described in 5.2.3.3.1 with each frame containing only one scan that contains the compressed data from only one band.

5.2.3.3.3 Frame header. The frame header specifies the source image characteristics, the components in the frame, the sampling factors for each component, and selects the quantization table to be used with each component. The format is shown in table IV with variable fields specified in table V for the different image types.

TABLE IV. Frame header.

Offset	Field Value	Field Name	length (bytes)	comments	
0	see table V	SOF_n	2	Start of frame. SOF_0 is used for "Baseline DCT sequential" mode when P=8. When P=12, SOF_1 must be used for "Extended DCT sequential, Huffman coding". Essentially, Baseline requires: sequential DCT, P=8; Huffman coding; 8-bit quantization tables; and no more than two sets of Huffman tables. Extended sequential allows: P=12, arithmetic coding; 16-bit quantization tables, and up to four sets of Huffman tables.	
2	see table V	L_f	2	Length of parameters = $(8+3N_f)$	
4	see table V	P	1	Sample precision, 8 or 12, (see SOF_n note)	
5	1-65535	Y	2	Number of lines (note 0 is not allowed)	
7	1-65535	X	2	Number of samples per line	
9	see table V	N_f	1	Number of components per frame, 1 or 3	
10	0	C_1	1	Component number = 0 (R or Y)	
11	see table V	H_1V_1	1	Horizontal & vertical sampling factors	
12	see table V	TQ_1	1	Quantization table selector	
13	1	C_2	1	Component number = 1 (G or Cb)	if $N_f = 3$
14	see table V	H_2V_2	1	Horizontal & vertical sampling factors	if $N_f = 3$
15	see table V	TQ_2	1	Quantization table selector	if $N_f = 3$
16	2	C_3	1	Component number = 2 (B or Cr)	if $N_f = 3$
17	see table V	H_3V_3	1	Horizontal & vertical sampling factors	if $N_f = 3$
18	see table V	TQ_3	1	Quantization table selector	if $N_f = 3$

TABLE V. Variable frame header fields.

Field Name	8-bit gray scale	12-bit gray scale	RGB color	YCbCr601 color	
SOF_n	0xFFC0 (SOF_0)	0xFFC1 (SOF_1)	0xFFC0 (SOF_0)	0xFFC0 (SOF_0)	
L_f	11	11	17	17	
P	8	12	8	8	
N_f	1	1	3	3	
C_1	0	0	0 (R)	0 (Y)	
H_1V_1	0 x 11	0 x 11	0 x 11	0x11, or 0x21, or 0x12, or 0x22	no subsampling (Cb, Cr) subsampled horiz. (Cb, Cr) subsampled vert. (Cb, Cr) subsampled horizontally & vertically
TQ_1	0	0	0	0	
C_2			1 (G)	1 (Cb)	
H_2V_2			0 x 11	0 x 11	
TQ_2			1	1	
C_3			2 (B)	2 (Cr)	
H_3V_3			0 x 11	0 x 11	
TQ_3			2	1	

5.2.3.3.4 Scan header. The scan header specifies which component(s) are contained in the scan and selects the entropy coding tables to be used with each component. The format is shown in table VI with variable fields specified in table VII for the different types.

TABLE VI. Scan header.

Offset	Field Value	Field Name	length (bytes)	comments	
0	0xFFDA	SOS	2	Start of scan.	
2	see table VII	L_s	2	Length of parameters = $(6+2N_s)$.	
4	see table VII	N_s	1	Number of components in scan, 1 or 3.	
5	0	Cs_1	1	Scan component selector (R or Y).	
6	see table VII	Td_1Ta_1	1	(DC, AC) entropy table selectors.	
7	1	Cs_2	1	Scan component selector (G or Cb).	if $N_s = 3$
8	see table VII	Td_2Ta_2	1	(DC, AC) entropy table selectors.	if $N_s = 3$
9	2	Cs_3	1	Scan component selector (B or Cr).	if $N_s = 3$
10	see table VII	Td_3Ta_3	1	(DC, AC) entropy table selectors.	if $N_s = 3$
7 or 11	0	S_s	1	Start of spectral selection = 0 (NA sequential DCT).	
8 or 12	63	S_e	1	End of spectral selection = 63 (NA sequential DCT).	
9 or 13	0x00	A_hA_l	1	Successive approximation bit positions (NA).	

TABLE VII. Variable scan header fields.

Field Name	8-bit gray scale	12-bit gray scale	YCbCr601 color	RGB color (interleaved)	RGB color (scan 1)	RGB color (scan 2)	RGB color (scan 3)
L_s	8	8	12	12	8	8	8
N_s	1	1	3	3	1	1	1
Cs_1	0	0	0 (Y)	0 (R)	0 (R)	1 (G)	2 (B)
Td_1Ta_1	0x00	0x00	0x00	0x00 or 0x11	0x00 or 0x11	0x00 or 0x11	0x00 or 0x11
Cs_2			1 (Cb)	1 (G)			
Td_2Ta_2			0x11	0x00 or 0x11			
Cs_3			2 (Cr)	2 (B)			
Td_3Ta_3			0x11	0x00 or 0x11			

5.2.3.3.5 Table-specification and miscellaneous marker segments. At the places indicated [tables/misc.] on figures 19, 20, and 21 any of the table-specification segments or miscellaneous marker segments specified in 5.2.3.3.5.1 - 5.2.3.3.5.5 may be present in any order and with no limit on the number of segments. If any table specifications occur in the compressed image data, they shall replace any defaults or previous specifications, and shall be used whenever the tables are required in the remaining scans in the frame. If a table specification occurs more than once for a given table in the compressed image data, each specification shall replace the previous specification.

5.2.3.3.5.1 Quantization table-specification. The quantization table segment format is shown in table VIII with variable fields specified in table IX for the different image types. Note that there may be one or more quantization tables specified per marker segment. For example, it is possible to specify the two tables used for YCbCr601 DCT compression with one or two DQT marker segments as shown in table IX.

TABLE VIII. Quantization table specification.

Offset	Field Value	Field Name	length (bytes)	comments	
0	0xFFDB	DQT	2	Define quantization table marker.	
2	see table IX	L_q	2	Length of parameters.	
4	see table IX	P_qT_q	1	Quantization table element precision. P_q specifies the precision of the Q_k values in table $\#T_q$. P_q value 0 indicates 8-bit Q_k values; value 1 indicates 16-bit Q_k values. P_q shall be zero for 8-bit sample precision P.	first table
5	see table IX	Q_k	64 or 128	Quantization table elements (64) in zig-zag order.	first table
	see table IX	P_qT_q	1	Quantization table element precision.	last table
	see table IX	Q_k	64 or 128	Quantization table elements (64) in zig-zag order.	last table

TABLE IX. Variable DQT segment fields.

Field Name	8/12-bit gray scale (8-bit tables)	12-bit gray scale (16-bit tables)	YCbCr601 color (2 tables)	YCbCr601 color (1 table)	RGB color (3 tables)	RGB color (1 table)	
L_q	67	131	132	67	197	67	
P_qT_q	0x00	0x10	0x00	0x00, or 0x01	0x00	0x00, or 0x01, or 0x02	first table
Q_k	1-255	1-65535	1-255	1-255	1-255	1-255	
P_qT_q			0x01		0x01		second table
Q_k			1-255		1-255		
P_qT_q					0x02		third table
Q_k					1-255		

5.2.3.3.5.2 Huffman table-specification. The Huffman table segment format is shown in table X with variable fields specified in table XI for the different image types. Note that there may be one or more Huffman tables specified per marker segment. For example, it is possible to specify the two tables used for YCbCr601 DCT compression with one or two DHT marker segments as shown in table XI.

TABLE X. Huffman table specification.

Offset	**Field Value**	**Field Name**	**length (bytes)**	**comments**
0	0xFFC4	DHT	2	Define Huffman table marker.
2	see table XI	L_h	2	Length of parameters.
4	see table XI	T_cT_h	1	T_c: Table class; 0=DC table; 1=AC table. T_h: Huffman table identifier (0-1 for baseline).
5	0-255	L_i	16	Number of codes of each length (BITS array).
21	0-255	$V_{i,j}$	see table XI	Symbols (HUFFVAL array).
	see table XI	T_cT_h	1	T_c: Table class; 0=DC table; 1=AC table. T_h: Huffman table identifier (0-1 for baseline).
	0-255	L_i	16	Number of codes of each length (BITS array).
	0-255	$V_{i,j}$	see table XI	Symbols (HUFFVAL array).

first table

first table

first table

last table

last table

last table

TABLE XI. Variable DHT fields.

Field Name	8-bit gray scale (1 DC/AC table)	12-bit gray scale (1 DC/AC table)	YCbCr601 color (2 DC/AC tables)	YCbCr601 color (1 DC/AC table)	RGB color (2 DC/AC tables)	RGB color (1 DC/AC table)	
L_h	210	278	418	210	418	210	
T_cT_h	0x00	0x00	0x00	0x00	0x00	0x00	DC table
# of $V_{i,j}$	12	16	12	12	12	12	
T_cT_h	0x10	0x10	0x10	0x10	0x10	0x10	AC table
# of $V_{i,j}$	162	226	162	162	162	162	
T_cT_h			0x01		0x01		DC table
# of $V_{i,j}$			12		12		
T_cT_h			0x11		0x11		AC table
# of $V_{i,j}$			162		162		

5.2.3.3.5.3 Restart interval definition. The restart interval definition segment format is shown in table XII. NITF requires that the restart interval be no more than the number of MCUs in a block-row.

TABLE XII. Restart interval definition.

Offset	Field Value	Field Name	length (bytes)	comments
0	0xFFDD	DRI	2	Define restart interval marker.
2	4	L_r	2	Length of parameters.
4	1-65535	R_i	2	Number of MCU in restart interval.

5.2.3.3.5.4 Comment segment. Use of the comment segment is optional for generators and may be ignored by interpreters. The segment structure is shown in table XIII.

TABLE XIII. Comment segment.

Offset	Field Value	Field Name	length (bytes)	comments
0	0xFFFE	COM	2	Comment marker.
2	2-65535	L_c	2	Segment length (2+length of comment).
4	0-255	Cm_1-Cm_{L_c-2}	L_c-2	Comment bytes.

5.2.3.3.5.5 Application data segment. JPEG defines an application data segment with the general structure in table XIV. Sixteen different application marker codes are defined: APP_0 - APP_f with corresponding values 0xFFE0 - 0xFFEF.

TABLE XIV. Application data segment.

Offset	Field Value	Field Name	length (bytes)	comments
0	0xFFF0- 0xFFEF	APP_n	2	Application data marker: APP_0 - APP_F.
2	2-65535	L_p	2	Segment length (2+length of application data).
4	0-255	AP_1-AP_{L_p-2}	L_p-2	Application data bytes.

5.2.3.3.5.5.1 NITF application data segment. NITF requires the use of an APP_6 application data segment. No other application data segments shall be present in the compressed data. The NITF application data segment shall immediately follow the first SOI marker in the Image Data Field. The NITF application data segment contains information which is needed by an interpreter but not supported by the ISO/CCITT JPEG format. Most of this information is also present in some fields of the NITF image subheader (COMRAT, IREPBAND, NBPP). The format is shown in table XV.

TABLE XV. NITF application data segment.

Offset	Field Value	Field Name	length (bytes)	comments
0	0xFFE6	APP_6	2	NITF application data marker.
2	25	L_p	2	Segment length (2+length of application data).
4	0x4E49 0x5446 0x00	Identifier	5	Zero terminated string: "NITF."
9	0x0200	Version	2	Version number. The most significant byte is used for major revisions, the least significant byte for minor revisions. Version 2.00 is the current revision level.
11	0x42, 0x50 or 0x53	IMODE	1	Image format. Three values are defined at this time. 'B' - IMODE=B 'P' - IMODE=P 'S' - IMODE=S
12	1-9999	H	2	Number of image blocks per row.
14	1-9999	V	2	Number of image blocks per column.
16	0-1	Image Color	1	Original image color representation. Two values are defined at this time. 0 - monochrome 1 - RGB
17	1-16	Image Bits	1	Original image sample precision.
18	0-99	Image Class	1	Image data class (0-99). One value is defined at this time. 0 - general purpose
19	1-29	JPEG Process	1	JPEG coding process. The values for this field are defined to be consistent with ISO DIS 10918-2. Two values are defined at this time. 1 - baseline sequential DCT, Huffman coding, 8-bit sample precision 4 - extended sequential DCT, Huffman coding, 12-bit sample precision
20	0-5	Quality	1	Image default quantization tables used. Quality values 1-5 select specific tables (in conjunction with the Image Class, Stream Color, and Stream Bits). The value 0 indicates no defaults and all quantization tables must then be present in the JPEG stream.

TABLE XV. NITF application data segment - Continued.

Offset	Field Value	Field Name	length (bytes)	comments
21	0-2	Stream Color	1	Compressed color representation. Three values are defined at this time. 0 - monochrome 1 - RGB 2 - YCbCr601
22	8 or 12	Stream Bits	1	Compressed image sample precision.
23	1	Horizontal Filtering	1	This field specifies the filtering used in the horizontal direction prior to subsampling the chrominance samples. One value is defined at this time. 1 - Centered samples, [1/2, 1/2] filter
24	1	Vertical Filtering	1	This field specifies the filtering used in the vertical direction prior to subsampling the chrominance samples. One value is defined at this time. 1 - Centered samples, [1/2, 1/2] filter
25	0	Flags	2	Reserved for future use.

5.2.4 Encoding procedure with marker codes. Figure 15 illustrates the overall encoding procedure when the marker codes are added to the entropy-coded data segments.

5.2.5 Decoding procedure with marker codes. Figure 16 illustrates the overall decoding procedure when the marker codes are added to the entropy-coded data segments.

5.2.5.1 Quantization tables. If the DQT marker is not in the compressed data, then information from the COMRAT field in the NITF image subheader (defined in MIL-STD-2500) shall be interpreted to determine the appropriate default table(s). If the DQT marker is in the compressed data, then this table specification shall take precedence over any defaults specified in the COMRAT field.

5.2.5.2 Huffman tables. If the DHT marker is not in the compressed data, then the default Huffman table, from Appendix B, for this image data type, image sample precision, and image color shall be used. If the DHT marker is in the compressed data, then this table specification shall take precedence over any defaults.

5.3 Progressive DCT-based JPEG mode. (Effectivity 5)

5.4 Hierarchical JPEG mode. (Effectivity 6)

5.5 Lossless JPEG mode. (Effectivity 2)

5.6 Region of interest encoding and decoding processes. (Effectivity 3)

6. NOTES

(This section contains general or explanatory information that may be helpful but is not mandatory).

6.1 Critical data. The JPEG marker segments (frame header, scan header, DQT, DHT, DRI, APP_6) are critical data. Corruption will result if the data is lost.

6.2 Input sample precision not 8 or 12. For DCT-based compression, the input sample precision for coding must be 8 or 12. To code source image data with a different sample precision, the data first must be converted. One-bit source image data should use alternate NITFS compression algorithms. Two and three-bit data should not be compressed. If the source sample precision is 4-7 bits, then the data should be converted to 8 bits. If the source sample precision is 9 - 11 bits or more than 12 bits, then the data should be converted to 12 bits. The conversion can be accomplished by least significant bit padding, interpolation, or by look-up tables. Least significant bit padding refers to left shifting the original data so that it occupies the most significant bits in the new 8- or 12-bit sample. The decoder can optionally convert the data to the original sample precision using the ABPP field in the NITF image subheader if required. The recommended method is to convert other than M-bit imagery into M-bit imagery using the following equation where M equals the number of bits required by the compression algorithm.

N = number of bits-per-pixel
P_N = N-bit pixel value
P_M = M-bit pixel value

$$P_M = \frac{2^M - 1}{2^N - 1} P_N$$

6.3 Use of restart intervals. Restart intervals introduce some overhead into the data stream to provide a level of error protection. A "smart decoder" will detect a transmission error as an invalid data stream during the decoding process, then skip forward looking for the next restart marker code to resynchronize. A tradeoff exists between the amount of overhead and the level of protection obtained. Neglecting the effects of packet size and error handling in the communications protocol, errors can be contained to a single restart interval. The overhead introduced by each restart interval is 20 bits, on average, for Huffman coding. Compressing an 8-bit monochrome image at 10:1 generates 50 bits on average per 8x8 block. If the number of blocks-per-interval is 40 or more, the overhead is one percent or less. An image with 512 samples-per-line consists of 64 blocks per block-row, so that the NITF required maximum restart interval is 64, resulting in less than one percent of overhead. For noisier environments, DRI 32 would contain the effects of each error into a half block-row.

6.4 Definition of effectivity. Some of the capabilities specified in this document are not required as of the issue date of the document. All such capabilities are marked with effectivity numbers (for example, Effectivity 1). Each effectivity number will be replaced by a specific date in subsequent releases of this document.

6.4.1 Effectivity 1 - Tables.

a. APPENDIX A 30.3 12-bit gray scale default quantization tables. (Effectivity 1)

b. APPENDIX A 30.4 24-bit color default quantization tables. (Effectivity 1)

c. APPENDIX B 30.3 12-bit gray scale BITS and HUFFVAL tables. (Effectivity 1)

d. APPENDIX B 30.4 24-bit color BITS and HUFFVAL tables. (Effectivity1)

6.4.2 Effectivity 2 - Lossless coding.

a. 4.10 Lossless coding. (Effectivity 2)

b. 5.1.2 Lossless encoding and decoding processes. (Effectivity 2)

c. 5.5 Lossless JPEG mode. (Effectivity 2)

6.4.3 Effectivity 3 - Region of interest coding.

a. 4.16 Region of interest coding. This standard allows different regions within a single image to be compressed at different rates (Type (Effectivity 3)), with resulting variation in reconstructed image quality, based on their judged relative importance.

b. 5.6 Region of interest encoding and decoding processes. (Effectivity 3)

6.4.4 Effectivity 4 - Multiple band images.

a. 5.1.1.2 Multiple-component imagery. (Effectivity 4)

b. 5.1.1.2.2 Multispectral imagery. (Effectivity 4)

6.4.5 Effectivity 5 - Progressive coding.

a. 4.11.2 Progressive DCT-based mode. (Effectivity 5)

b. 5.3 Progressive DCT-based JPEG mode. (Effectivity 5)

6.4.6 Effectivity 6 - Hierarchical coding.

a. 4.11.3 Hierarchical mode. (Effectivity 6)

b. 5.4 Hierarchical JPEG mode. (Effectivity 6)

6.4.7 Effectivity 7 - Arithmetic coding.

a. 4.12 Entropy coding alternatives. Two alternative entropy coding procedures are specified: Huffman coding (Type 1, Type 2, Type 3) and arithmetic coding (Type (Effectivity 7)).

b. 5.2.2.5.2 Arithmetic coding. (Effectivity 7)

6.4.8 Effectivity 8 - Defense Information Systems Network (DISN).

a. DISA/JIEO Circular 9008

6.5 Subject term (key word) listing.

BWC
Compression algorithm
Continuous Tone Imagery
DCT
Discrete Cosine Transforms
Gray scale imagery
Huffman coding
Image compression
Quantization Matrices
Secondary Imagery Dissemination Systems
SIDS

APPENDIX A

DEFAULT QUANTIZATION TABLES

10. GENERAL

10.1 Scope. This appendix is a mandatory part of the standard. The information it contains is intended for compliance.

20. APPLICABLE DOCUMENTS

20.1 Government documents. This section is not applicable to this appendix.

30. DEFINITIONS

30.1 Definitions used in this appendix. For purposes of this appendix, the definitions are at the beginning of this document.

40. GENERAL REQUIREMENTS

40.1 Default quantization tables. For each combination of image data type, image sample precision, and image color, there are five default quantization tables allowing images to be coded at five different quality levels. Quality level 5 (Q5) reconstructed image data has the highest fidelity to the source image data but achieves the least compression. Levels 4, 3, 2, and 1 trade the reconstructed fidelity for higher compression, with Q1 resulting in the most compression. The table are listed in the subsequent sections, and all are listed in zig-zag order (see 5.1.1.6).

40.2 Eight-bit gray scale default quantization tables.

TABLE A-I. Eight-bit gray scale quantization tables (data type 0).

zig-zag index	**Q1**	**Q2**	**Q3**	**Q4**	**Q5**
0 (=DC)	8	8	8	8	4
1	72	36	10	7	4
2	72	36	10	7	4
3	72	36	10	7	4

TABLE A-I. Eight-bit gray scale quantization tables (data type 0) - Continued.

zig-zag index	**Q1**	**Q2**	**Q3**	**Q4**	**Q5**
4	72	36	10	7	4
5	72	36	10	7	4
6	72	36	10	7	4
7	72	36	10	7	4
8	72	36	10	7	4
9	72	36	10	7	4
10	78	39	11	8	4
11	74	37	10	7	4
12	76	38	11	8	4
13	74	37	10	7	4
14	78	39	11	8	4
15	89	45	13	9	5
16	81	41	11	8	5
17	84	42	12	8	5
18	84	42	12	8	5
19	81	41	11	8	5
20	89	45	13	9	5
21	106	53	15	11	6

TABLE A-I. Eight-bit gray scale quantization tables (data type 0) - Continued.

zig-zag index	**Q1**	**Q2**	**Q3**	**Q4**	**Q5**
22	93	47	13	9	5
23	94	47	13	9	5
24	99	50	14	10	6
25	94	47	13	9	5
26	93	47	13	9	5
27	106	53	15	11	6
28	129	65	18	13	7
29	111	56	16	11	6
30	108	54	15	11	6
31	116	59	16	12	6
32	116	59	16	12	6
33	108	54	15	11	6
34	111	56	16	11	6
35	129	65	18	13	7
36	135	68	19	14	8
37	128	64	18	13	7
38	136	69	19	14	8
39	145	73	21	15	8

TABLE A-I. Eight-bit gray scale quantization tables (data type 0) - Continued.

zig-zag index	Q1	Q2	Q3	Q4	Q5
40	136	69	19	14	8
41	128	64	18	13	7
42	135	68	19	14	8
43	155	78	22	16	9
44	160	81	23	16	9
45	177	89	25	18	10
46	177	89	25	18	10
47	160	81	23	16	9
48	155	78	22	16	9
49	193	98	27	20	11
50	213	108	30	22	12
51	228	115	32	23	13
52	213	108	30	22	12
53	193	98	27	20	11
54	255	130	36	26	14
55	255	144	40	29	16
56	255	144	40	29	16
57	255	130	36	26	14

TABLE A-I. Eight-bit gray scale quantization tables (data type 0) - Continued.

zig-zag index	Q1	Q2	Q3	Q4	Q5
58	255	178	50	36	20
59	255	190	53	38	21
60	255	178	50	36	20
61	255	243	68	49	27
62	255	243	68	49	27
63	255	255	91	65	36

NOTE: Additional quantizer tables will be added in future versions of this standard for specific image data types (visual, SAR, IR, fingerprints, maps) as soon as technical work codifies requirements and validates fitness for use.

40.3 12-bit gray scale default quantization tables. (Effectivity 1)

40.4 24-bit color default quantization tables. (Effectivity 1)

APPENDIX B

DEFAULT HUFFMAN TABLES

10. GENERAL

10.1 Scope. This appendix is a mandatory part of the standard. The information it contains is intended for compliance.

20. APPLICABLE DOCUMENTS

20.1 Applicable documents. This section is not applicable to this appendix.

30. DEFINITIONS

30.1 Definitions used in this appendix. For purposes of this appendix, the definitions are at the beginning of this document.

40. GENERAL REQUIREMENTS

40.1 Default Huffman tables.

40.1.1 Default general purpose BITS and HUFFVAL tables. For each combination of image data type, image sample precision, and image color there are default Huffman tables. The Huffman codes are generated from two tables: BITS and HUFFVAL. BITS is a list of 16 eight-bit values defining the number of Huffman codes of each size, one through 16. HUFFVAL is a list of eight-bit symbols, one symbol for each code in increasing code length order. All tables in the subsequent sections should be read from left to right.

40.1.2 Eight-bit gray scale BITS and HUFFVAL tables. Note that the eight-bit default tables are identical to the example luminance tables in the ISO/CCITT JPEG standard.

TABLE B-I. Eight-bit gray scale DC BITS table.

0	1	5	1	1	1	1	1
1	0	0	0	0	0	0	0

TABLE B-II. Eight-bit gray scale AC BITS table.

0	2	1	3	3	2	4	3
5	5	4	4	0	0	1	125

TABLE B-III. Eight-bit gray scale DC HUFFVAL table.

0	1	2	3	4	5	6	7
8	9	10	11				

TABLE B-IV. Eight-bit gray scale AC HUFFVAL table.

0x01	0x02	0x03	0x00	0x04	0x11	0x05	0x12
0x21	0x31	0x41	0x06	0x13	0x51	0x61	0x07
0x22	0x71	0x14	0x32	0x81	0x91	0xA1	0x08
0x23	0x42	0xB1	0xC1	0x15	0x52	0xD1	0xF0
0x24	0x33	0x62	0x72	0x82	0x09	0x0A	0x16
0x17	0x18	0x19	0x1A	0x25	0x26	0x27	0x28
0x29	0x2A	0x34	0x35	0x36	0x37	0x38	0x39
0x3A	0x43	0x44	0x45	0x46	0x47	0x48	0x49
0x4A	0x53	0x54	0x55	0x56	0x57	0x58	0x59
0x5A	0x63	0x64	0x65	0x66	0x67	0x68	0x69
0x6A	0x73	0x74	0x75	0x76	0x77	0x78	0x79
0x7A	0x83	0x84	0x85	0x86	0x87	0x88	0x89

TABLE B-IV. Eight-bit gray scale AC HUFFVAL table - Continued.

0x8A	0x92	0x93	0x94	0x95	0x96	0x97	0x98
0x99	0x9A	0xA2	0xA3	0xA4	0xA5	0xA6	0xA7
0xA8	0xA9	0xAA	0xB2	0xB3	0xB4	0xB5	0xB6
0xB7	0xB8	0xB9	0xBA	0xC2	0xC3	0xC4	0xC5
0xC6	0xC7	0xC8	0xC9	0xCA	0xD2	0xD3	0xD4
0xD5	0xD6	0xD7	0xD8	0xD9	0xDA	0xE1	0xE2
0xE3	0xE4	0xE5	0xE6	0xE7	0xE8	0xE9	0xEA
0xF1	0xF2	0xF3	0xF4	0xF5	0xF6	0xF7	0xF8
0xF9	0xFA						

40.1.3 12-bit gray scale BITS and HUFFVAL tables. (Effectivity 1)

40.1.4 24-bit color BITS and HUFFVAL tables. (Effectivity 1)

APPENDIX C

GENERATING CUSTOM HUFFMAN TABLE SPECIFICATIONS

10. GENERAL

10.1 Scope. This appendix is not a mandatory part of the standard. The information it contains is intended for guidance when custom Huffman tables are used to compress.

20. APPLICABLE DOCUMENTS

20.1 Applicable documents. This section is not applicable to this appendix.

30. DEFINITIONS

30.1 Definitions used in this appendix. For purposes of this appendix, the definitions are at the beginning of this document.

40. GENERAL REQUIREMENTS

40.1 Generating custom Huffman table specifications.

40.2 Custom Huffman table generation. The Huffman codes are generated from two tables, BITS and HUFFVAL. This appendix specifies how to generate BITS and HUFFVAL. Appendix D specifies a method of code generation, so that, given BITS and HUFFVAL, the associated codes are defined uniquely. BITS is a list of 16 eight-bit values defining the number of Huffman codes of each size, one through 16. HUFFVAL is a list of 8-bit symbols, one symbol for each code in increasing code length order.

40.3 Gathering statistics. This procedure is only needed at the encoder. The required statistics are FREQ(V), the frequency of occurrence of symbol V. In JPEG, there are never more than 256 symbols; therefore FREQ(V) is collected for V = 0 to 255. FREQ values for unused symbols are defined to be zero. This procedure must be repeated for the DC and the AC coefficients for each set of components to be coded using this custom Huffman coding table.

40.4 <u>Generating the Huffman code sizes (CODESIZE)</u>. In JPEG there are never more than 256 symbols, and the code lengths are limited to 16 by design. A procedure for determining the Huffman code sizes (CODESIZE) for a set of up to 256 symbols is shown on figure C-1. [21] Three vectors are defined for this procedure:

a. FREQ(V): frequency of occurrence of symbol V

b. CODESIZE (V): code size of symbol V

c. OTHERS (V): index to nest symbol in chain of all symbols in current branch of code tree

where V goes from 0 to 256.

Before starting the procedure, the values for FREQ are collected for V = 0 to 255, and the FREQ value for V=256 is set to 1 to reserve one code point. FREQ values for unused symbols are defined to be zero. In addition, the entries in CODESIZE are set to 0, and the indices in OTHERS are set to -1, the value which terminates a chain of indices. Reserving one code point guarantees that no code word can ever be all '1' bits.

The search for the entry with the least value of FREQ(V) selects the largest value of V with the least value of FREQ(V) greater than zero.

The procedure "Find V1 for the least value of FREQ(V1) > 0" always selects the value with the largest value of V1 when more than one V1 with the same frequency occurs. The reserved code point then is guaranteed to be in the longest code word category.

[21] Ibid, p. 176.

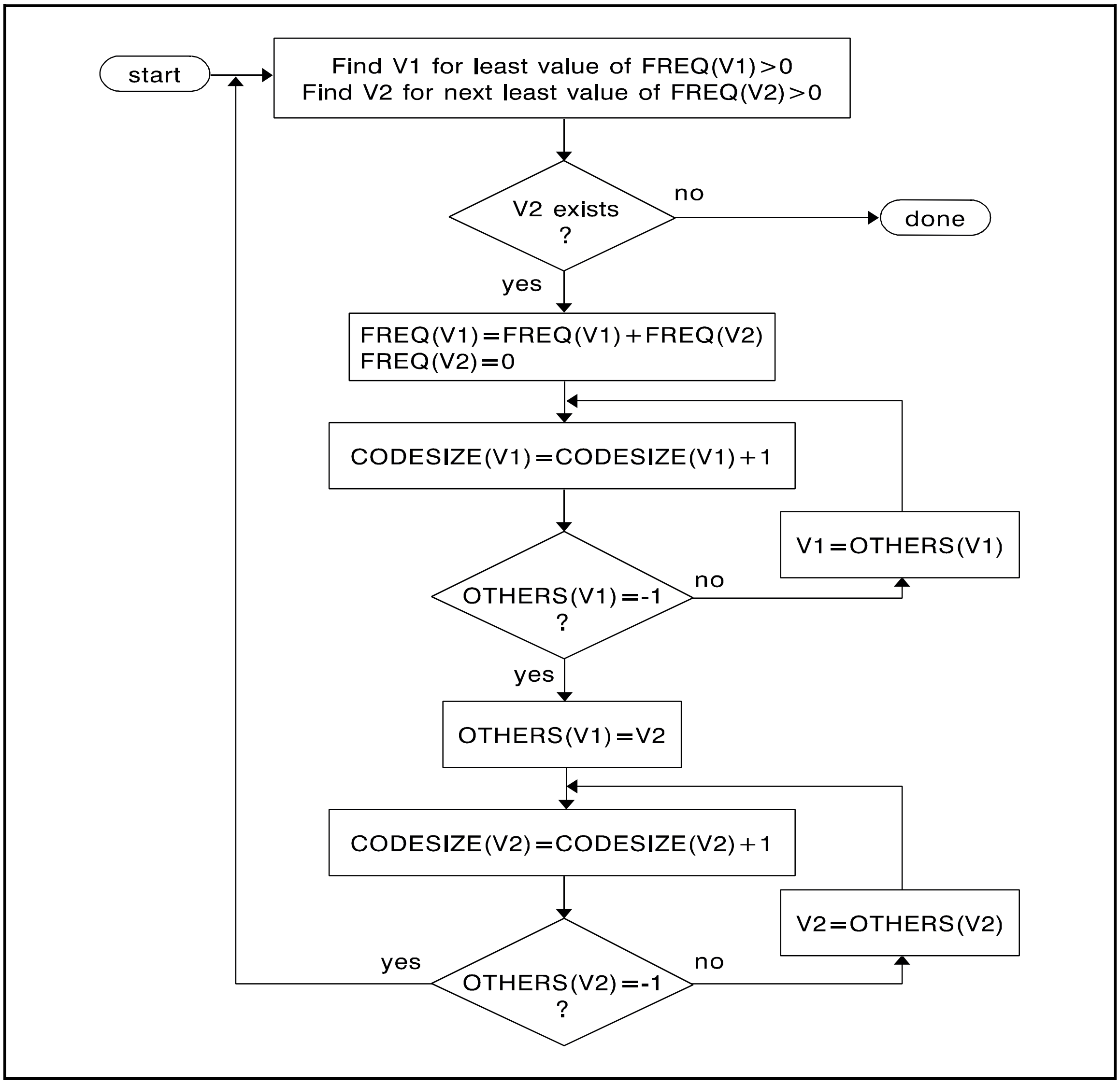

FIGURE C-1. Procedure to find Huffman code sizes.

40.5 Generating the number of codes of each size (BITS). Once CODESIZE has been obtained, the number of codes of each length (BITS(I); I=1,2,...) is obtained using the procedure on figure C-2. [22] The counts in BITS are zero at the start of the procedure. Note that until the next procedure is complete, BITS may have more than the 16 entries allowed in this standard.

[22]Ibid, p. 177.

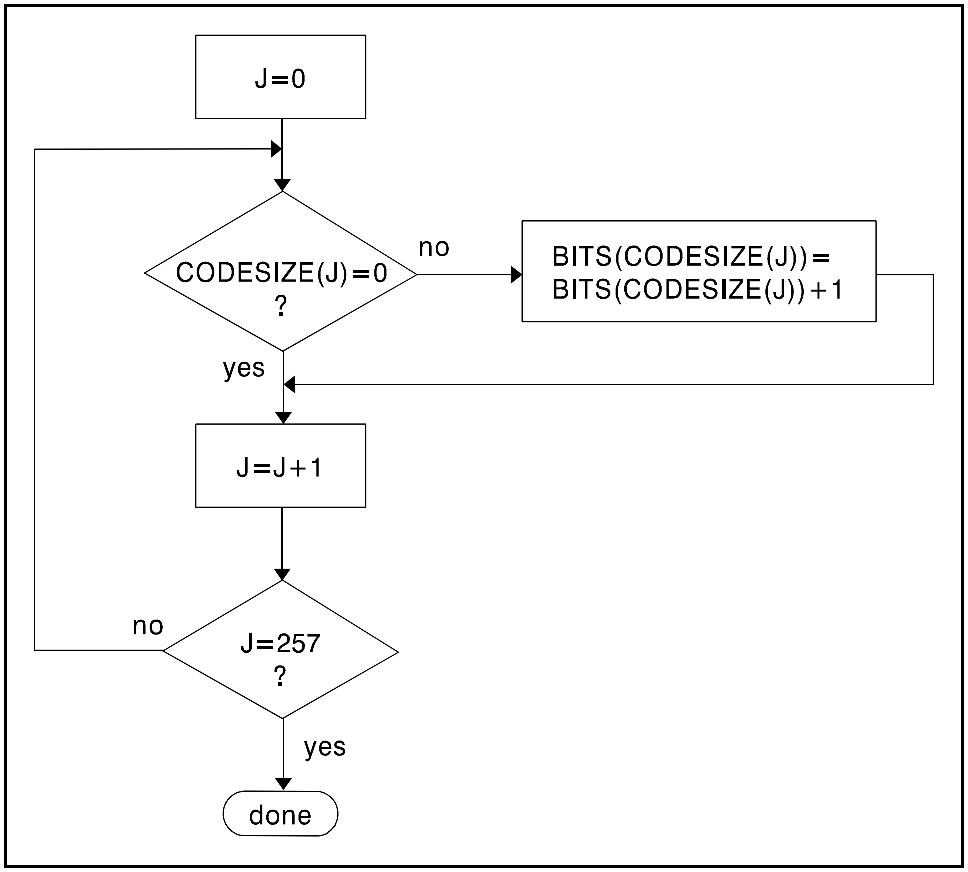

FIGURE C-2. Procedure to find the number of codes of each size.

40.5.1 Limiting the code lengths to 16 bits. Figure C-3 [23] gives the procedure for adjusting the BITS list so that no code is longer than 16 bits. The procedure assumes that the probabilities are distributed in a way such that code lengths greater than 32 bits never occur so that the input is BITS(I) whereas I=1,2,…,32. Since symbols are paired for the longest Huffman code, the symbols are removed from this length category two at a time. The prefix for the pair (which is one bit shorter) is allocated to one of the pair, then a code word from the next shortest code length (skipping the prefix length) is converted into a prefix for two code words one bit longer. After the BITS list is reduced to a maximum code length of 16 bits, the last step removes the reserved code point from the code length count.

[23] Ibid, p. 178.

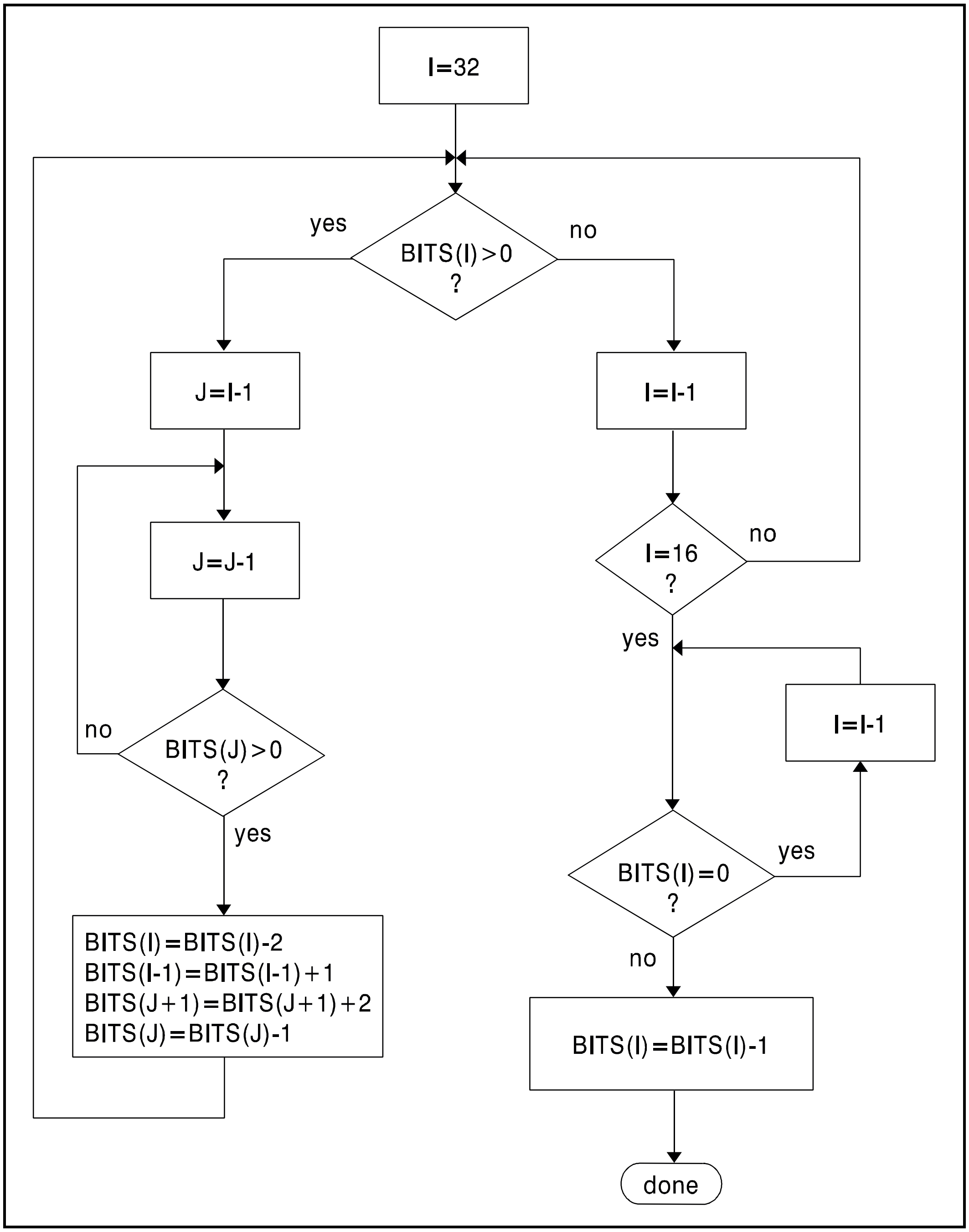

FIGURE C-3. Procedure for limiting code lengths to 16 bits.

40.6 Sorting the input values according to code size (HUFFVAL). The input values are sorted according to code size as shown on figure C-4 [24] HUFFVAL is the list containing the input values associated with each code word, in order of increasing code length.

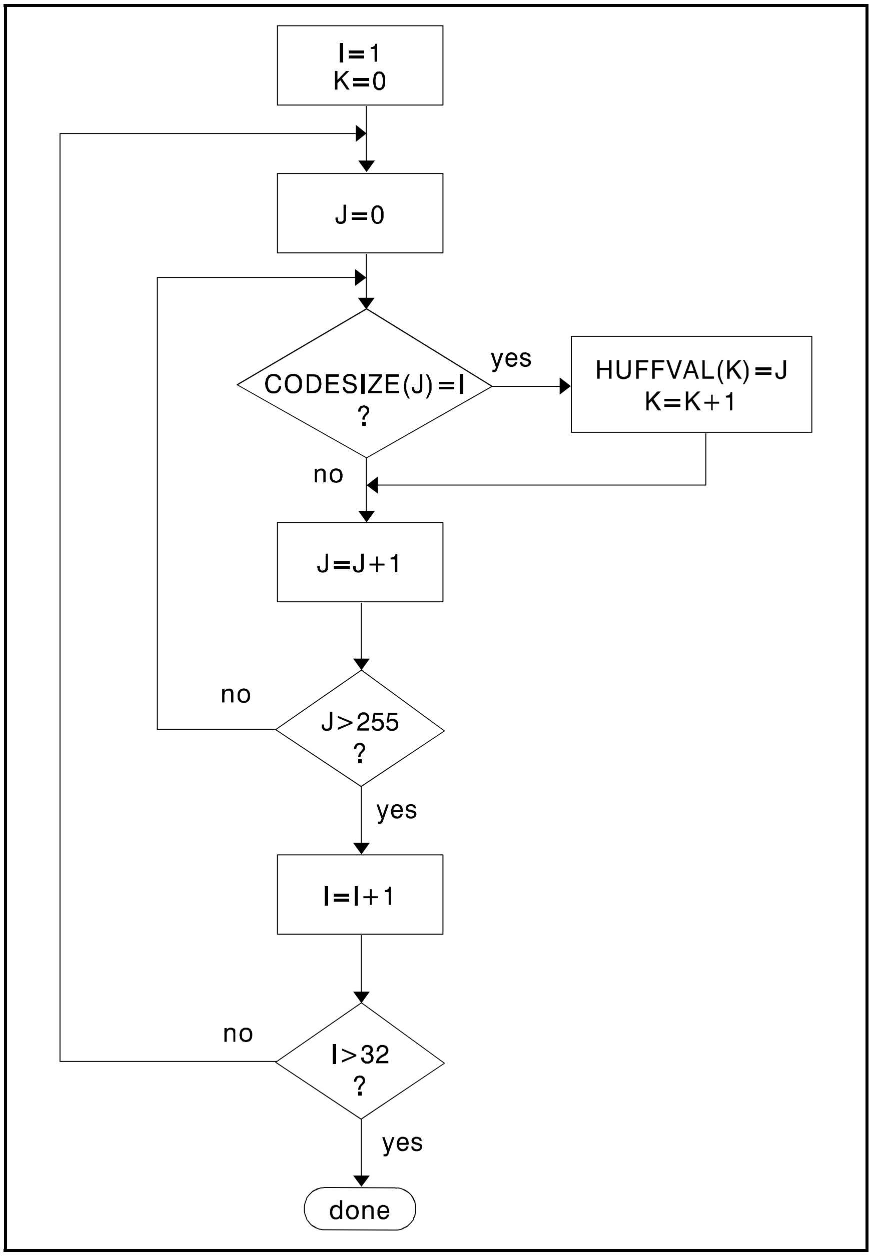

FIGURE C-4. Sorting of input values according to code size.

[24]Ibid, p. 179.

APPENDIX D

BUILDING HUFFMAN TABLES FROM THE BITS/HUFFVAL SPECIFICATION

10. GENERAL

10.1 Scope. This appendix is a mandatory part of the standard. The information it contains is intended for compliance.

20. APPLICABLE DOCUMENTS

20.1 Applicable documents. This section is not applicable to this appendix.

30. DEFINITIONS

30.1 Definitions used in this appendix. For purposes of this appendix, the definitions are at the beginning of this document.

40. GENERAL REQUIREMENTS

40.1 Building Huffman tables from the BITS/HUFFVAL specification. The Huffman coding tables, EHUFCO and EHUFSI, are generated from {BITS(I);I = 1, …,16} and {HUFFVAL(K); K = 0, …, (LASTK-1)} where LASTK is determined on figure D-1. [25]

[25] Ibid, p. 78.

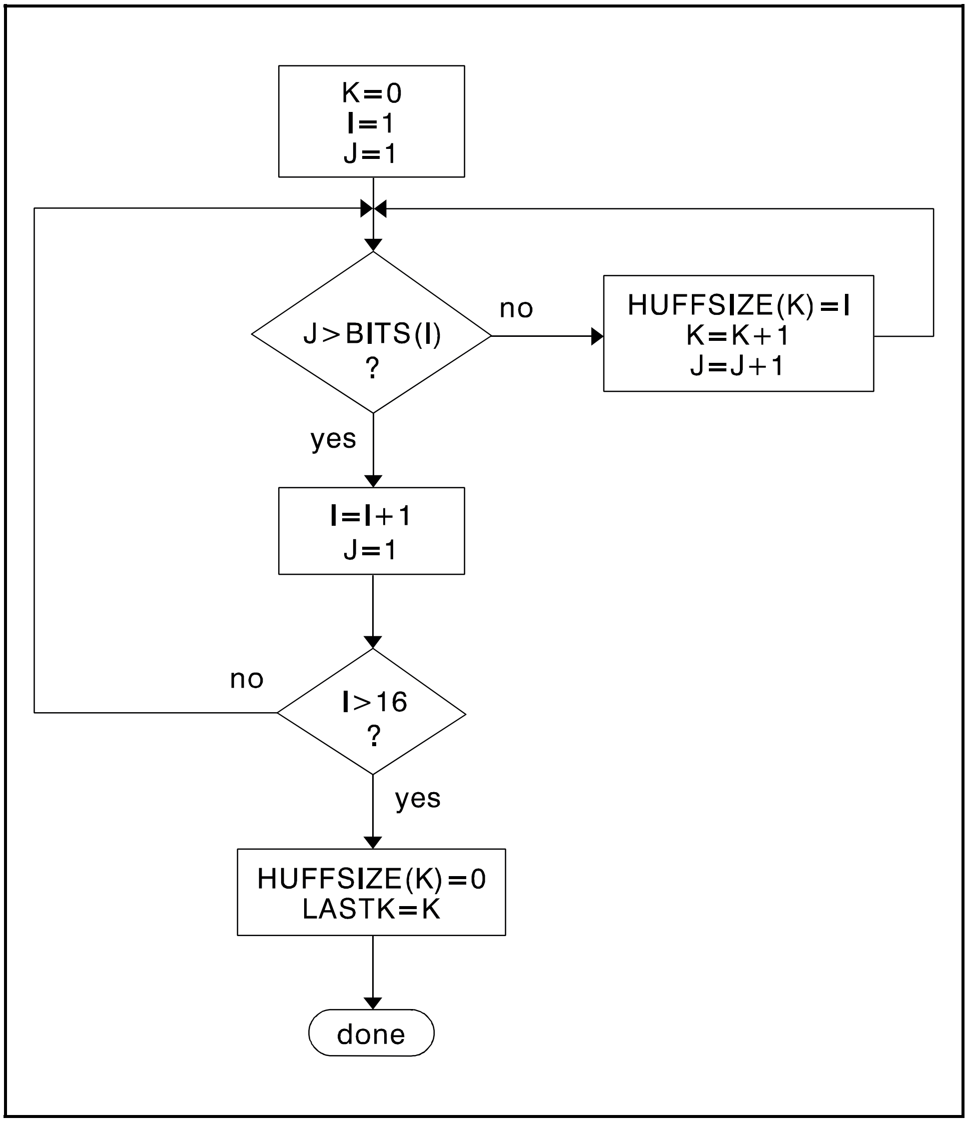

FIGURE D-1. Generation of HUFFSIZE, a vector of Huffman code sizes.

40.2 Custom Huffman table generation. Two intermediate vectors, {HUFFSIZE(K); K = 0, …., LASTK} and {HUFFCODE(K); K = 0, …., (LASTK - 1)} must first be generated.

40.3 Generating HUFFSIZE. From BITS, the vector HUFFSIZE is generated containing the code lengths (size) corresponding to each symbol in HUFFVAL. HUFFSIZE is generated by the procedure on figure D-1.

40.4 Generating HUFFCODE. Next, a Huffman code table, HUFFCODE, containing a code for each symbol in HUFFVAL, is generated by the procedure on figure D-2. [26] To facilitate this, the procedure on figure D-1 set HUFFSIZE (LASTK) = 0, where LASTK is one more than the largest valid value for K. The notation (SL CODE 1) on figure D-2 indicates a shift left of CODE by one bit in position.

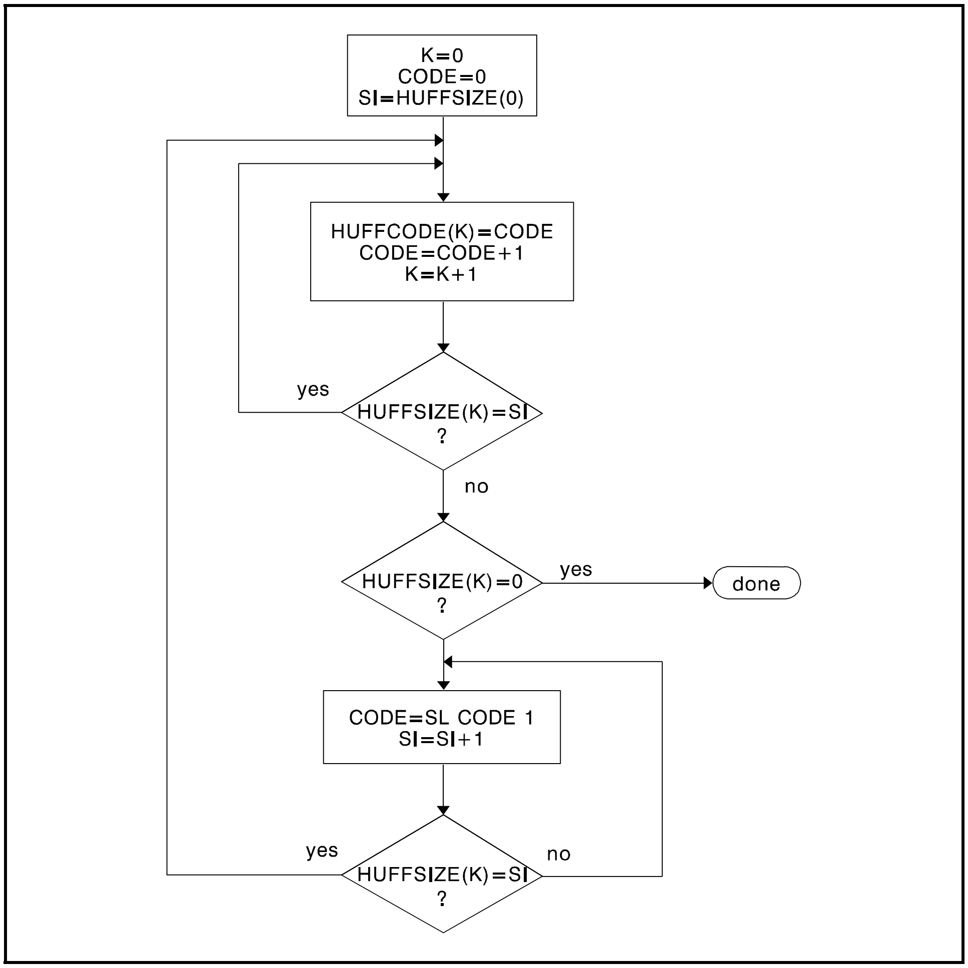

FIGURE D-2. Generation of the Huffman codes: HUFFCODE.

[26]Ibid, p. 79.

40.5 Generating EHUFCO and EHUFSI. Two tables, HUFFCODE and HUFFSIZE, have now been initialized. However, the entries in the tables are ordered according to increasing code length rather than by symbol as required for encoding. The encoding procedure code tables, EHUFCO and EHUFSI, are created by ordering HUFFCODE and HUFFSIZE according to the symbol values in HUFFVAL. Figure D-3 [27] illustrates this reordering procedure.

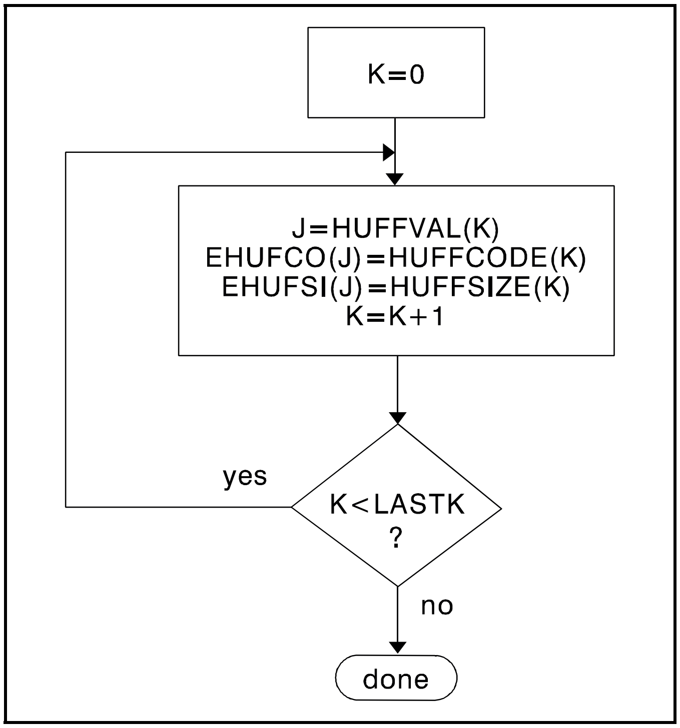

FIGURE D-3. Ordering to generate the encoding procedure code tables.

40.6 Default EHUFCO and EHUFSI tables for eight-bit gray scale. The EHUFCO and EHUFSI tables are shown below corresponding to the eight-bit gray scale default general purpose BITS and HUFFVAL tables specified in Appendix B, 40.1.2. Table D-I shows the values for DC coding. Table D-II shows the values for AC coding. Note: For AC coding, not all values of V are valid as described in 5.2.2.5.1.4. For example, 11 - 14 are only valid for 12-bit sample data, and 15 is always invalid.

[27] Ibid, p. 80.

TABLE D-I. EHUFCO and EHUFSI for eight-bit gray scale DC Huffman Codes.

VALUE, V	EHUFCO(V)	EHUFSI(V)	Binary Huffman Code
0	0x0000	2	00
1	0x0002	3	010
2	0x0003	3	011
3	0x0004	3	100
4	0x0005	3	101
5	0x0006	3	110
6	0x000E	4	1110
7	0x001E	5	11110
8	0x003E	6	111110
9	0x007E	7	1111110
10	0x00FE	8	11111110
11	0x01FE	9	111111110

TABLE D-II. EHUFCO and EHUFSI for eight-bit gray scale AC Huffman Codes.

VALUE, V	EHUFCO(V)	EHUFSI(V)	Binary Huffman Code
0	0x000A	4	1010
1	0x0000	2	00
2	0x0001	2	01
3	0x0004	3	100

TABLE D-II. EHUFCO and EHUFSI for eight-bit gray scale AC Huffman Codes - Continued.

VALUE, V	EHUFCO(V)	EHUFSI(V)	Binary Huffman Code
4	0x000B	4	1011
5	0x001A	5	11010
6	0x0078	7	1111000
7	0x00F8	8	11111000
8	0x03F6	10	1111110110
9	0xFF82	16	1111111110000010
10	0xFF83	16	1111111110000011
11	not valid	not valid	not valid
12	not valid	not valid	not valid
13	not valid	not valid	not valid
14	not valid	not valid	not valid
15	not valid	not valid	not valid
16	not valid	not valid	not valid
17	0x000C	4	1100
18	0x001B	5	11011
19	0x0079	7	1111001
20	0x01F6	9	111110110
21	0x07F6	11	11111110110
22	0xFF84	16	1111111110000100

TABLE D-II. EHUFCO and EHUFSI for eight-bit gray scale AC Huffman Codes - Continued.

VALUE, V	EHUFCO(V)	EHUFSI(V)	Binary Huffman Code
23	0xFF85	16	1111111110000101
24	0xFF86	16	1111111110000110
25	0xFF87	16	1111111110000111
26	0xFF88	16	1111111110001000
27	not valid	not valid	not valid
28	not valid	not valid	not valid
29	not valid	not valid	not valid
30	not valid	not valid	not valid
31	not valid	not valid	not valid
32	not valid	not valid	not valid
33	0x001C	5	11100
34	0x00F9	8	11111001
35	0x03F7	10	1111110111
36	0x0FF4	12	111111110100
37	0xFF89	16	1111111110001001
38	0xFF8A	16	1111111110001010
39	0xFF8B	16	1111111110001011
40	0xFF8C	16	1111111110001100
41	0xFF8D	16	1111111110001101

TABLE D-II. EHUFCO and EHUFSI for eight-bit gray scale AC Huffman Codes - Continued.

VALUE, V	EHUFCO(V)	EHUFSI(V)	Binary Huffman Code
42	0xFF8E	16	1111111110001110
43	not valid	not valid	not valid
44	not valid	not valid	not valid
45	not valid	not valid	not valid
46	not valid	not valid	not valid
47	not valid	not valid	not valid
48	not valid	not valid	not valid
49	0x003A	6	111010
50	0x01F7	9	111110111
51	0x0FF5	12	111111110101
52	0xFF8F	16	1111111110001111
53	0xFF90	16	1111111110010000
54	0xFF91	16	1111111110010001
55	0xFF92	16	1111111110010010
56	0xFF93	16	1111111110010011
57	0xFF94	16	1111111110010100
58	0xFF95	16	1111111110010101
59	not valid	not valid	not valid
60	not valid	not valid	not valid

TABLE D-II. EHUFCO and EHUFSI for eight-bit gray scale AC Huffman Codes - Continued.

VALUE, V	EHUFCO(V)	EHUFSI(V)	Binary Huffman Code
61	not valid	not valid	not valid
62	not valid	not valid	not valid
63	not valid	not valid	not valid
64	not valid	not valid	not valid
65	0x003B	6	111011
66	0x03F8	10	1111111000
67	0xFF96	16	1111111110010110
68	0xFF97	16	1111111110010111
69	0xFF98	16	1111111110011000
70	0xFF99	16	1111111110011001
71	0xFF9A	16	1111111110011010
72	0xFF9B	16	1111111110011011
73	0xFF9C	16	1111111110011100
74	0xFF9D	16	1111111110011101
75	not valid	not valid	not valid
76	not valid	not valid	not valid
77	not valid	not valid	not valid
78	not valid	not valid	not valid
79	not valid	not valid	not valid

TABLE D-II. EHUFCO and EHUFSI for eight-bit gray scale AC Huffman Codes - Continued.

VALUE, V	EHUFCO(V)	EHUFSI(V)	Binary Huffman Code
80	not valid	not valid	not valid
81	0x007A	7	1111010
82	0x07F7	11	11111110111
83	0xFF9E	16	1111111110011110
84	0xFF9F	16	1111111110011111
85	0xFFA0	16	1111111110100000
86	0xFFA1	16	1111111110100001
87	0xFFA2	16	1111111110100010
88	0xFFA3	16	1111111110100011
89	0xFFA4	16	1111111110100100
90	0xFFA5	16	1111111110100101
91	not valid	not valid	not valid
92	not valid	not valid	not valid
93	not valid	not valid	not valid
94	not valid	not valid	not valid
95	not valid	not valid	not valid
96	not valid	not valid	not valid
97	0x007B	7	1111011
98	0x0FF6	12	111111110110

TABLE D-II. EHUFCO and EHUFSI for eight-bit gray scale AC Huffman Codes - Continued.

VALUE, V	EHUFCO(V)	EHUFSI(V)	Binary Huffman Code
99	0xFFA6	16	1111111110100110
100	0xFFA7	16	1111111110100111
101	0xFFA8	16	1111111110101000
102	0xFFA9	16	1111111110101001
103	0xFFAA	16	1111111110101010
104	0xFFAB	16	1111111110101011
105	0xFFAC	16	1111111110101100
106	0xFFAD	16	1111111110101101
107	not valid	not valid	not valid
108	not valid	not valid	not valid
109	not valid	not valid	not valid
110	not valid	not valid	not valid
111	not valid	not valid	not valid
112	not valid	not valid	not valid
113	0x00FA	8	11111010
114	0x0FF7	12	111111110111
115	0xFFAE	16	1111111110101110
116	0xFFAF	16	1111111110101111
117	0xFFB0	16	1111111110110000

TABLE D-II. EHUFCO and EHUFSI for eight-bit gray scale AC Huffman Codes - Continued.

VALUE, V	EHUFCO(V)	EHUFSI(V)	Binary Huffman Code
118	0xFFB1	16	1111111110110001
119	0xFFB2	16	1111111110110010
120	0xFFB3	16	1111111110110011
121	0xFFB4	16	1111111110110100
122	0xFFB5	16	1111111110110101
123	not valid	not valid	not valid
124	not valid	not valid	not valid
125	not valid	not valid	not valid
126	not valid	not valid	not valid
127	not valid	not valid	not valid
128	not valid	not valid	not valid
129	0x01F8	9	111111000
130	0x7FC0	15	111111111000000
131	0xFFB6	16	1111111110110110
132	0xFFB7	16	1111111110110111
133	0xFFB8	16	1111111110111000
134	0xFFB9	16	1111111110111001
135	0xFFBA	16	1111111110111010
136	0xFFBB	16	1111111110111011

TABLE D-II. EHUFCO and EHUFSI for eight-bit gray scale AC Huffman Codes - Continued.

VALUE, V	EHUFCO(V)	EHUFSI(V)	Binary Huffman Code
137	0xFFBC	16	1111111110111100
138	0xFFBD	16	1111111110111101
139	not valid	not valid	not valid
140	not valid	not valid	not valid
141	not valid	not valid	not valid
142	not valid	not valid	not valid
143	not valid	not valid	not valid
144	not valid	not valid	not valid
145	0x01F9	9	111111001
146	0xFFBE	16	1111111110111110
147	0xFFBF	16	1111111110111111
148	0xFFC0	16	1111111111000000
149	0xFFC1	16	1111111111000001
150	0xFFC2	16	1111111111000010
151	0xFFC3	16	1111111111000011
152	0xFFC4	16	1111111111000100
153	0xFFC5	16	1111111111000101
154	0xFFC6	16	1111111111000110
155	not valid	not valid	not valid

TABLE D-II. EHUFCO and EHUFSI for eight-bit gray scale AC Huffman Codes - Continued.

VALUE, V	EHUFCO(V)	EHUFSI(V)	Binary Huffman Code
156	not valid	not valid	not valid
157	not valid	not valid	not valid
158	not valid	not valid	not valid
159	not valid	not valid	not valid
160	not valid	not valid	not valid
161	0x01FA	9	111111010
162	0xFFC7	16	1111111111000111
163	0xFFC8	16	1111111111001000
164	0xFFC9	16	1111111111001001
165	0xFFCA	16	1111111111001010
166	0xFFCB	16	1111111111001011
167	0xFFCC	16	1111111111001100
168	0xFFCD	16	1111111111001101
169	0xFFCE	16	1111111111001110
170	0xFFCF	16	1111111111001111
171	not valid	not valid	not valid
172	not valid	not valid	not valid
173	not valid	not valid	not valid
174	not valid	not valid	not valid

TABLE D-II. EHUFCO and EHUFSI for eight-bit gray scale AC Huffman Codes - Continued.

VALUE, V	EHUFCO(V)	EHUFSI(V)	Binary Huffman Code
175	not valid	not valid	not valid
176	not valid	not valid	not valid
177	0x03F9	10	1111111001
178	0xFFD0	16	1111111111010000
179	0xFFD1	16	1111111111010001
180	0xFFD2	16	1111111111010010
181	0xFFD3	16	1111111111010011
182	0xFFD4	16	1111111111010100
183	0xFFD5	16	1111111111010101
184	0xFFD6	16	1111111111010110
185	0xFFD7	16	1111111111010111
186	0xFFD8	16	1111111111011000
187	not valid	not valid	not valid
188	not valid	not valid	not valid
189	not valid	not valid	not valid
190	not valid	not valid	not valid
191	not valid	not valid	not valid
192	not valid	not valid	not valid
193	0x03FA	10	1111111010

TABLE D-II. EHUFCO and EHUFSI for eight-bit gray scale AC Huffman Codes - Continued.

VALUE, V	EHUFCO(V)	EHUFSI(V)	Binary Huffman Code
194	0xFFD9	16	1111111111011001
195	0xFFDA	16	1111111111011010
196	0xFFDB	16	1111111111011011
197	0xFFDC	16	1111111111011100
198	0xFFDD	16	1111111111011101
199	0xFFDE	16	1111111111011110
200	0xFFDF	16	1111111111011111
201	0xFFE0	16	1111111111100000
202	0xFFE1	16	1111111111100001
203	not valid	not valid	not valid
204	not valid	not valid	not valid
205	not valid	not valid	not valid
206	not valid	not valid	not valid
207	not valid	not valid	not valid
208	not valid	not valid	not valid
209	0x07F8	11	11111111000
210	0xFFE2	16	1111111111100010
211	0xFFE3	16	1111111111100011
212	0xFFE4	16	1111111111100100

TABLE D-II. EHUFCO and EHUFSI for eight-bit gray scale AC Huffman Codes - Continued.

VALUE, V	EHUFCO(V)	EHUFSI(V)	Binary Huffman Code
213	0xFFE5	16	1111111111100101
214	0xFFE6	16	1111111111100110
215	0xFFE7	16	1111111111100111
216	0xFFE8	16	1111111111101000
217	0xFFE9	16	1111111111101001
218	0xFFEA	16	1111111111101010
219	not valid	not valid	not valid
220	not valid	not valid	not valid
221	not valid	not valid	not valid
222	not valid	not valid	not valid
223	not valid	not valid	not valid
224	not valid	not valid	not valid
225	0xFFEB	16	1111111111101011
226	0xFFEC	16	1111111111101100
227	0xFFED	16	1111111111101101
228	0xFFEE	16	1111111111101110
229	0xFFEF	16	1111111111101111
230	0xFFF0	16	1111111111110000
231	0xFFF1	16	1111111111110001

TABLE D-II. EHUFCO and EHUFSI for eight-bit gray scale AC Huffman Codes - Continued.

VALUE, V	EHUFCO(V)	EHUFSI(V)	Binary Huffman Code
232	0xFFF2	16	1111111111110010
233	0xFFF3	16	1111111111110011
234	0xFFF4	16	1111111111110100
235	not valid	not valid	not valid
236	not valid	not valid	not valid
237	not valid	not valid	not valid
238	not valid	not valid	not valid
239	not valid	not valid	not valid
240	0x07F9	11	11111111001
241	0xFFF5	16	1111111111110101
242	0xFFF6	16	1111111111110110
243	0xFFF7	16	1111111111110111
244	0xFFF8	16	1111111111111000
245	0xFFF9	16	1111111111111001
246	0xFFFA	16	1111111111111010
247	0xFFFB	16	1111111111111011
248	0xFFFC	16	1111111111111100
249	0xFFFD	16	1111111111111101
250	0xFFFE	16	1111111111111110

TABLE D-II. EHUFCO and EHUFSI for eight-bit gray scale AC Huffman Codes - Continued.

CONCLUDING MATERIAL

Custodians:
Army - SC
Navy - OM
Air Force - 02
Misc - DC

Review activities:
OASD - SO, DO, HP, IR
Army - AM, AR, MI, TM, MD, CE, SC, IE, ET, AC, PT
Navy - OM
Air Force - 02
DLA - DH
Misc - NS, MP, DI, NA

User activities:
OASD - SO, DO, HP, IR
Army - AM, AR, MI, TM, MD, CE, SC, IE, ET, AC, PT
Navy - OM
Air Force - 02
DLA - DH
Misc - NS, MP, DI, NA

Civil agency coordinating activities:
USDA - AFS, APS
COM - NIST
DOE
EPA
GPO
HHS - NIH
DOI - BLM, GES, MIN
DOT - CGCT

Preparing activity:
Misc - DC

Agent:
Not applicable

(Project TCSS-1980)

STANDARDIZATION DOCUMENT IMPROVEMENT PROPOSAL

INSTRUCTIONS

1. The preparing activity must complete blocks 1,2, 3, and 8. In block 1, both the document number and revision letter should be given.
2. The submitter of this form must complete blocks 4, 5, 6, and 7.
3. The preparing activity must provide a reply within 30 days from receipt of the form.

NOTE: This form may not be used to request copies of documents, nor to request waivers, or clarification of requirements on current contracts. Comments submitted on this form do not constitute or imply authorization to waive any portion of the referenced document(s) or to amend contractual requirements.

I RECOMMEND A CHANGE:	1. DOCUMENT NUMBER MIL-STD-188-198A	2. DOCUMENT DATE (YYMMDD) 15 December 1993

3. DOCUMENT TITLE JPEG IMAGE COMPRESSION FOR THE NITFS

4. NATURE OF CHANGE *(Identify paragraph number and include proposed rewrite, if possible. Attach extra sheets as needed.)*

5. REASON FOR RECOMMENDATION

6. SUBMITTER

a. NAME *(Last, First, Middle Initial)* Vlastakis, Emanuel	b. ORGANIZATION
c. ADDRESS *(Include Zip Code)* Bldg. 286 Ft. Monmouth, NJ 07703-5613	d. TELEPHONE *(Include Area Code)* (1) Commercial (2) AUTOVON *(If applicable)*

7. DATE SUBMITTED *(YYMMDD)*

8. PREPARING ACTIVITY DEFENSE INFORMATION SYSTEMS AGENCY (DISA)

a. NAME DISA/JIEO/CFS/TBCF	b. TELEPHONE *(Include Area Code)* (1) Commercial (2) AUTOVON
c. ADDRESS *(Include Zip Code)* 11440 ISAAC NEWTON SQUARE, NORTH RESTON, VA 22090-5006	**IF YOU DO NOT RECEIVE A REPLY WITHIN 45 DAYS, CONTACT:** Defense Quality and Standardization Office 5203 Leesburg Pike, Suite 1403, Falls Church, VA 22041-3466 Telephone (703) 756-2340 AUTOVON 289-2340