ALLIED
ENGINEERING
DOCUMENTATION
PUBLICATION

AEDP-2
(Edition 1)

NATO Intelligence, Surveillance, and Reconnaissance (ISR) Interoperability Architecture (NIIA)

VOLUME 3: NIIA Technical Guidance

AEDP-2
(Edition 1)

SEPTEMBER 2005

NORTH ATLANTIC TREATY ORGANIZATION

NATO STANDARDIZATION AGENCY (NSA)

NATO LETTER OF PROMULGATION

September 2005

1. AEDP-2 (Edition 1) - NATO INTELLIGENCE, SURVEILLANCE, AND RECONNAISSANCE (ISR) INTEROPERABILITY ARCHITECTURE (NIIA) VOLUME 3: NIIA TECHNICAL GUIDANCE - is an NATO UNCLASSIFIED publication.

2. AEDP-2 (Edition 1) is effective upon receipt.

J. MAJ
Brigadier General, POL(A)
Director, NSA

RECORD OF CHANGES

Change Date	Date entered	Effective Date	By Whom Entered

FOREWARD

The NATO Intelligence, Surveillance, and Reconnaissance (ISR) Interoperability Architecture (NIIA) provides the basis for the technical aspects of an architecture that provides interoperability between NATO nations' ISR systems. It is recognized that a complete architecture requires a technical view, a systems view, and an operational view to be complete. However, the systems and operational views are dependant on the specific scenario, with the systems involved determined by the participating nations, and the operational view defining how the various systems are actually interconnected. While a acquisition group such as Air Group IV could theorize hypothetical scenarios and generate the systems and operational views based on those hypothetical scenarios, it is more important to focus on the technical issues of providing the interconnectivity options within national and NATO-owned systems, and leave the operations planning to the military community.

This AEDP provides the technical and management guidance for implementing the NIIA in ISR systems. It is divided into four volumes. Volume 1 provides the introduction and explanation of the technical architecture. Volume 2 contains guidance for managing the NIIA, specifically, the configuration management and test and certification guidance to the NIIA Custodians. Volume 3 is the technical guidance relevant to multiple parts of the architecture. Finally, Volume 4 provides terms and definitions. The four volumes are published as separate documents due to the large size of each volume.

This volume provides technical guidance for those topics that span more than one STANAG. Topics included in this document have general relevance to multiple parts of the architecture. Those technical topics that relate to only one standard are included in the AEDP associated with that particular STANAG.

In addition to AEDP-2, users of the NIIA should obtain copies of each of the STANAGs incorporated into the architecture. These STANAGs provide the key interface standards needed to provide the systems interoperability. Many of the STANAGs also have separate Implementation Guides for the standard, published as separate AEDPs. In addition, specific guidance on sanitization and declassification of advanced memory recording systems (solid state and advanced disk arrays) is provided in a separate document for ease of dissemination. This document is AEDP-3. A complete list of the documents included in the set of STANAGs/AEDPs is in Annex A of Volume 1 of this document.

Questions or comments on this document can be provided to either the Secretary of Air Group IV or the Custodian. Correspondence to the Secretary should be addressed to: Secretary, Air Group IV; Air Armaments Section, International Staff; HQ NATO; B-1110, Brussels, Belgium (telephone: +32-2-707-4291; telefax: +32-2-707-4103). Correspondence to the Custodian should be addressed to: Custodian, AEDP-2; SAF/AQIJ; 1060 Air Force Pentagon; Washington D.C. 20330-1060; United States (telephone +1 703-588-2669; telefax: +1 703-588-1340).

Table of Contents

Executive Summary

The NATO Intelligence, Surveillance, and Reconnaissance (ISR) Interoperability Architecture (NIIA) defines the overall structure of the elements of the Intelligence, Surveillance, and Reconnaissance (ISR) community. The intent of this document is to provide the context for the standards developed by NATO Air Group IV, as well as commercial and international standards that are applicable to the ISR mission. It should be noted that while the original NIIA document was a description of an imagery-only architecture, the work of Air Group IV has suggested that the scope of the Group's activities should include other sources of intelligence.

A description of operational environment, including a number of variations on the basic data flow, is provided. This description includes notional task flow descriptions and timelines. When examining the ISR data flow, it is noted what interfaces will be exercised during the two primary ISR Integration Working Group (ISRIWG) demonstrations. These two demonstrations, one to show interoperability at the input to the ground/exploitation system by using the NATO Advanced Data Storage Interface, and one to show interoperability at the output of the ground/exploitation system by using the NATO Secondary Imagery Format and NATO Standard Imagery Library Interface.

Each of the currently ratified and developmental standards developed under Air Group IV is also reviewed. Each standard is discussed in terms of its application and use. Other documents needed to complete the architecture are also discussed.

Finally, the actual architecture is introduced and discussed. Four levels of interoperability, as defined in NATO interoperability publications, are reviewed. They are:

- Degree 1: Unstructured Data Exchange. Involves the exchange of human-interpretable unstructured data such as the free text found in operational estimates, analysis and papers.
- Degree 2: Structured Data Exchange. Involves the exchange of human-interpretable structured data intended for manual and/or automated handling, but requires manual compilation, receipt and/or message dispatch.
- Degree 3: Seamless Sharing of Data. Involves the automated sharing of data amongst systems based on a common exchange model.
- Degree 4: Seamless Sharing of Information. An extension of degree 3 to the universal interpretation of information through data processing based on co-operating applications.

It should be noted that the objective of the NIIA is to achieve interoperability at Degree 2, with some specific interfaces achieving Degree 3. Degree 4 can be considered a long-term objective, but it was determined that lower degrees of interoperability should not be delayed in favour of ultimately achieving a higher degree. Degree 2 interoperability is a significant accomplishment, and will provide a high level of capability to NATO and coalition forces. Higher degrees of interoperability will be addressed once degree 2 is achieved and demonstrated.

Finally, a review of the standards as they fit into the architecture is performed. The standards are mapped against the International Standards Organisation (ISO) 7-Layer

Interface Model. This mapping is performed to identify how the standards fit together and include commercial and international standards, as well as to identify the holes in the architecture that must be addressed by the activities of the ISRIWG and its subordinate groups. During the analysis, it was determined that most interfaces are adequately defined, and those requiring additional standardization can be filled with existing commercial and/or international standards. The key issue remaining is the multitude of choices provided by many of the standards, thereby allowing multiple

implementations that would not be interoperable. It will be important to develop interface profiles that define the specific choices within each standard, thereby ensuring interoperability.

In summary, it is noted that the ISR architecture is applicable across all levels of NATO and coalition operations, including both Article 5 (war operations) and non-Article 5 (peacekeeping, peacemaking, etc.) campaigns. Finally, while it is recognised that the standards are complete, there is a large volume of support documentation, including configuration management plans, test and certification plans, implementation guidance, and acquisition guidance, that are available to the community. It is recommended that STANAG custodians consolidate this documentation into a single volume to accompany each standard. The accepted form of this volume is as an Allied Engineering Documentation Publication (AEDP). Combining the AEDPs with this document and the standards forms the complete NIIA definition and documentation set.

1.0 INTRODUCTION

1.0.1 This volume provides technical guidance on the implementation of the NIIA with guidance that applies to more than one STANAG. Technical implementation guidance applicable to only a single STANAG is contained in the implementation guidance for that STANAG and is published in the respective AEDP.

1.1 Scope

1.1.1 The technical implementation guidance in this document is that information applicable to multiple STANAGs or general enough in nature to be considered applicable to the overall architecture.

2.0 VOLUME CONTENTS

2.0.1 This volume is intended to contain a variety of topics. As additional technical implementation issues are identified, they will be added to this document. Each topic is included in a separate annex. The annexes that are included are as follows:

- Annex A: Image Compression Implementation Guidance – This Annex includes guidance on implementing the image compression techniques used in the NIIA. There are multiple techniques available for the NIIA community. Each technique is discussed in separate appendices to the Annex.
- Annex B: File Naming Conventions – This Annex will identify file naming conventions used in the NIIA. Specific applications may use unique conventions, but most will be able to identify the file attributes based on the conventions identified in this Annex.
- Annex C: Encryption in the NIIA – This Annex will review the use of encryption, particularly in wideband applications, as it applies to NIIA systems. Improperly used encryption will invalidate all other aspects of the NIIA.
- Annex D: Spectrum Management Guidelines for NIIA Systems – This Annex will identify the current constraints and provide general utilization guidelines for the electromagnetic spectrum in ISR scenarios.
- Annex E: Ground Station Coding and Capability Matrix – This Anne will provide guidelines for the identification of specific ground stations so that when used in an operational role, the force planners will know what capabilities they have available.

3.0 SUMMARY

3.0.1 Technical implementation guidance applicable to multiple portions of the NIIA is included herein. Guidance for individual STANAGs is included in the respective AEDP.

Image Compression Implementation Guidance

The image data formats include provisions for implementing data compression for the data in the image segments. For example, in the NATO Secondary Image format (NSIF), the compression is identified in the image subheader by referencing the Image Compression (IC) field. This field contains a valid code indicating the form of compression used in representing the image data. This is followed by the Compression Rate (COMRAT) field that provides an estimate of the ratio of the size of the original image file to the compressed file. Valid values for the IC field are, C1 to represent bi-level, C3 to represent JPEG, C4 to represent Vector Quantization, C5 to represent lossless JPEG, I1 to represent downsampled JPEG and NC to represent the image is not compressed. Also valid are M1, M3, M4, and M5 for compressed images, and NM for uncompressed images indicating an image that contains a Block Mask and/or a Pad Pixel Mask. (C6 and M6 are reserved values that will represent a future correlated multicomponent compression algorithm. C7 and M7 are reserved values that will represent a future complex SAR compression.) C8 and M8 are the values that will represent the ISO standard compression JPEG 2000. The definitions of the compression schemes associated with codes are as follows. C1/M1 is found in ITU-T T.4 AMD2. C3/M3 is found in MIL-STD-188-198A profile of ISO/IEC 10918-1 and ISO/IEC DIS 10918-3. C4/M4 is found in ISO/IEC IS 12087-5, and C5/M5 and I1 are found in U.S. NIMA Document N0106-98. (NOTE: C2 (ARIDPCM) is not valid in NSIF.) Values C6/M6 and C7/M7 will be defined as the respective standards become available. Values C8/M8 are defined in ISO/IEC IS 15444-1 (JPEG 2000) and the associated profile of JPEG 2000 tailored for the Basic Image Interchange Format (BIIF).

This document provides an overview as to how each compression algorithm is to be implemented. It is recommended that in addition to reviewing this implementation guidance, developers and users of image formats consult the latest version of the guiding standard identified above.

Each compression algorithm is treated separately in Appendices to this Annex. The Appendices are as follows:

Appendix 1	Bi-Level Compression
Appendix 2	JPEG
Appendix 3	Vector Quantization
Appendix 4	Lossless JPEG
Appendix 5	Downsampled JPEG
Appendix 6	JPEG 2000

Annex A, Appendix 1
Bi-Level Compression Implementation Considerations[1]

1.0 Introduction

NIIA systems using image data compressed using the bi-level facsimile compression will comply with the procedures specified by the International Telecommunications Union (ITU) International Telegraph and Telephone Consultative Committee (CCITT) Recommendation T.4 for Group 3 facsimile devices. No attempt has been made to discuss image scanning, communication, or printing systems.

1.1 Content

The ITU standard provides technical detail of the NIIA compression algorithm designated by the code C1 in the image compression field of the image subheader for bi-level images or overlays. It also provides the required run-length code tables for use in the NIIA.

1.2 Types of Operation

This standard establishes the requirements for the communication or interchange of image data in compressed form. The bi-level compression standard may be operated in one of three modes:

a. mode 1 - one-dimensional coding.
b. mode 2 - two-dimensional coding with standard vertical resolution, K = 2.
c. mode 3 - two-dimensional coding with higher vertical resolution, K = 4.

The corresponding modes are specified by 1D, 2DS, and 2DH, respectively, in the Compression Rate Code field of the NSIF file image subheader.

2.0 General Notes

2.1 NSIF Notes

2.1.1 Image Compression Field

The NSIF image compression field of the NSIF image header shall be specified as C1 for facsimile encoding for bi-level data.

2.1.2 NSIF Compression Rate Code Field

The parameter K shall be specified in the compression rate code field of the NSIF image header. The parameter K shall have a standard vertical resolution of K=2, and an optional higher resolution as specified by K=4.

2.2 Resolution

The maximum number of pixels allowable in each uncoded image line shall be 2560.

2.3 Fill

[1] This appendix contains excerpts from U.S. MIL-STD-188-196.

Optional fill may be governed by specific integration issues. For example, fill is inserted in T.4 facsimile devices to meet minimum line transmission time requirements as cited by section 3 of T.4.

2.4 Decoder

The decoder should be able to interpret fill data as well as recognize the two different vertical resolutions (K = 2 and K = 4).

Annex A, Appendix 2
JPEG Implementation Considerations[1]

1.0 Scope

Image data compressed using the JPEG image compression algorithm will comply with the procedures described in ISO/IEC IS 10918-1, *Digital Compression and Coding of Continuous-tone Still Images*.

1.1 Content

The standard and this appendix provides technical detail of the JPEG compression algorithm (designated by the code C3 in the Image Compression field of the NSIF file image subheader), 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 NSIF.

1.2 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.
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.

2.0 General Notes

2.1 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.

[1] This appendix contains excerpts from U.S. MIL-STD-188-198A.

2.2 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 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 NSIF 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 N
P_M = M-bit pixel value M

$$P_M = \frac{2_M - 1}{2_N - 1} * P_N$$

2.3 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 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.

2.4 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 tables are listed together, and all are listed in zig-zag order.

2.4.1 Eight-Bit Gray Scale Default Quantization Tables

Eight-Bit Gray Scale Default Quantization Tables					
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
4	72	36	10	7	4

Eight-Bit Gray Scale Default Quantization Tables					
zig-zag index	Q1	Q2	Q3	Q4	Q5
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
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
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

Eight-Bit Gray Scale Default Quantization Tables					
zig-zag index	Q1	Q2	Q3	Q4	Q5
54	255	130	36	26	14
55	255	144	40	29	16
56	255	144	40	29	16
57	255	130	36	26	14
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 document for specific image data types (visual, SAR, IR, fingerprints, maps) as soon as technical work codifies requirements and validates fitness for use.

2.5 Default Huffman Tables

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.

2.5.1 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.

Eight-Bit Gray Scale DC BITS Table							
0	1	5	1	1	1	1	1
1	0	0	0	0	0	0	0

Eight-Bit Gray Scale AC BITS Table							
0	2	1	3	3	2	4	3
5	5	4	4	0	0	1	125

Eight-Bit Gray Scale DC HUFFVAL Table							
0	1	2	3	4	5	6	7
8	9	10	11				

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
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						

2.6 Generating Custom Huffman Table Specifications

2.6.1 Custom Huffman Table Generation

The Huffman codes are generated from two tables, BITS and HUFFVAL. This appendix specifies how to generate BITS and HUFFVAL. Section 2.7 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.

2.6.2 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.

2.6.3 Generating the Huffman Code Sizes (CODESIZE)

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 A-2-1. Three vectors are defined for this procedure:

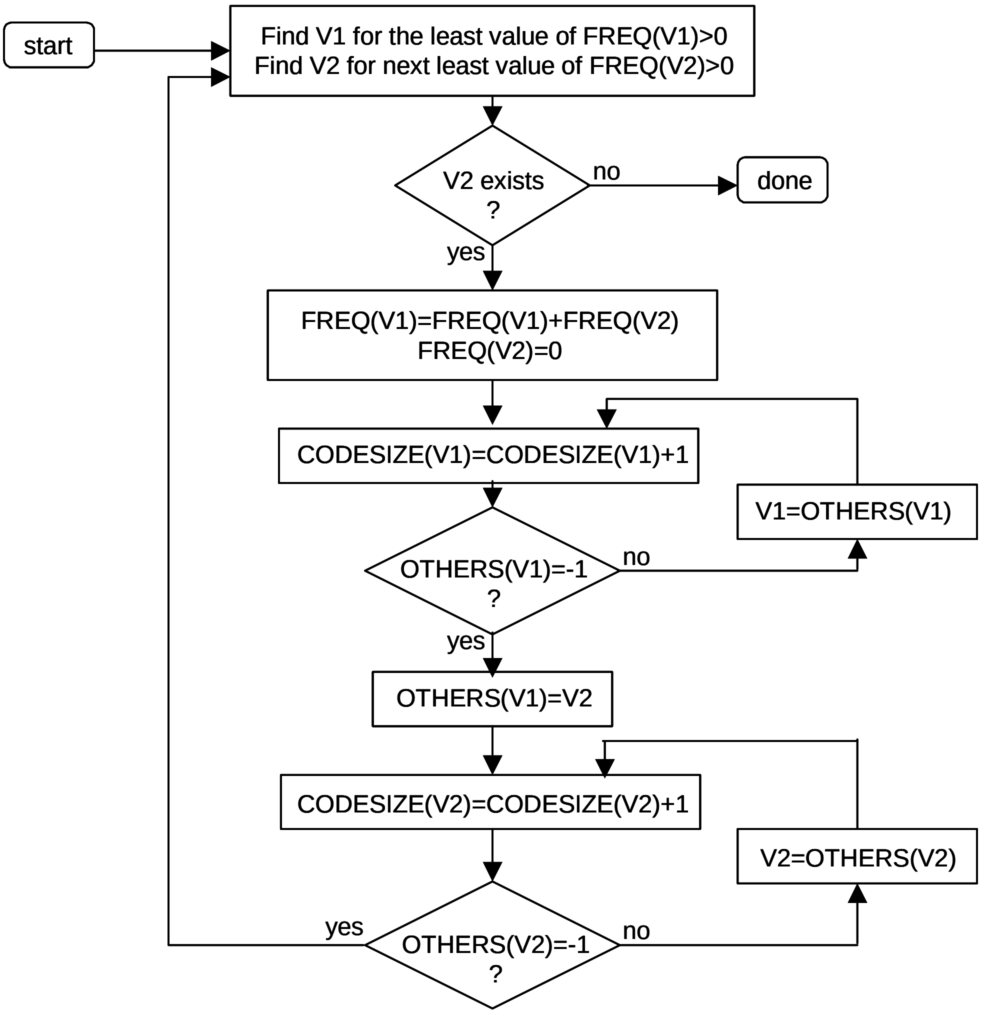

Figure A-2-1 - Procedure to Find Huffman Code Sizes

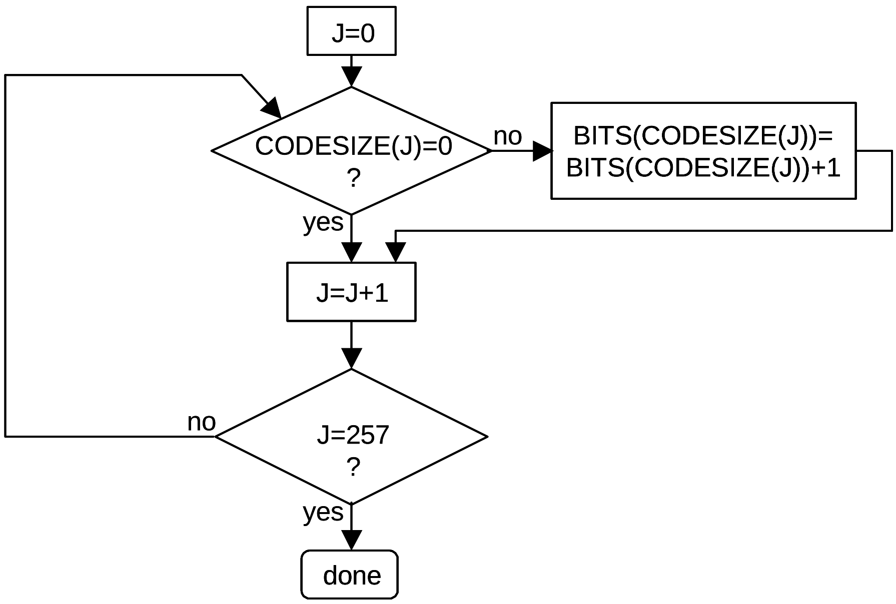

Figure A-2-2 - Procedure to Find Number of Codes of Each Size

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.

2.6.4 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 A-2-2. The 22 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.

2.6.5 Limiting the Code Lengths to 16 Bits

Figure A-2-3 gives the procedure for adjusting the BITS 23 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

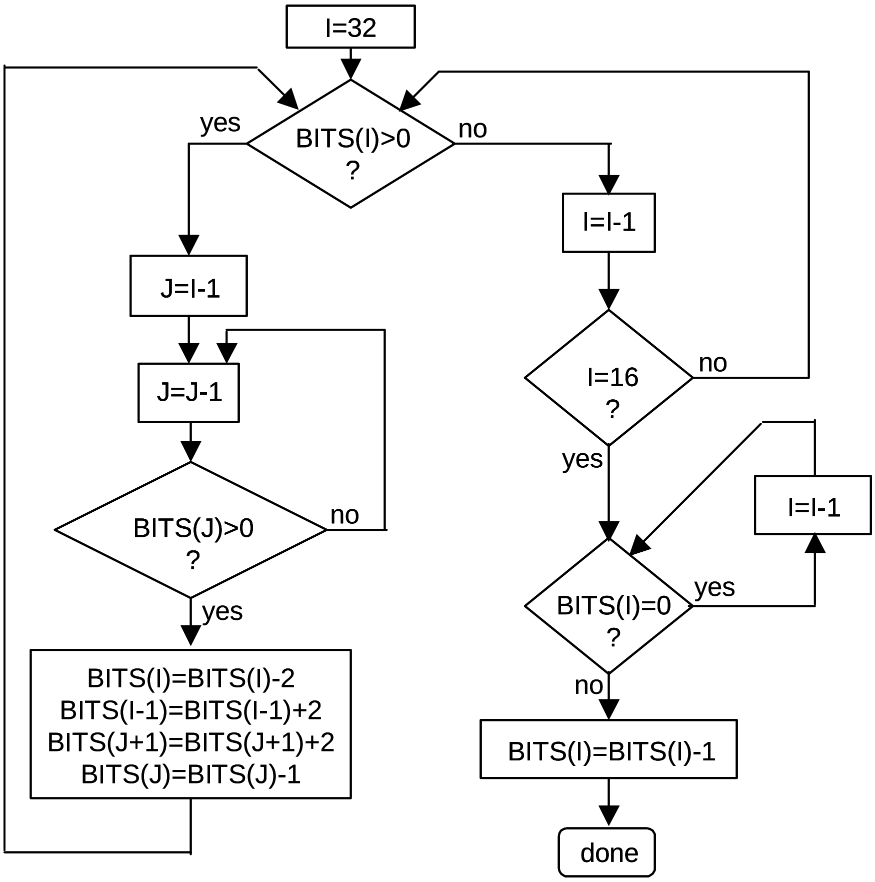

Figure A-2-3 - Procedure for Limiting Code Lengths to 16 Bits

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.

2.6.6 Sorting the Input Values According to Code Size (HUFFVAL)

The input values are sorted according to code size as shown on figure A-2-4. HUFFVAL is the list containing the input values 24 associated with each code word, in order of increasing code length.

Annex A, Appendix 3
Vector Quantization Implementation Considerations[1]

1.0 Vector Quantized Data

Vector quantization (VQ) is a structuring algorithm chosen for use on multiband, color, and gray scale raster scanned maps and imagery because it provides predictable, rapid image reconstruction results. All information required for reconstruction of a VQ file is contained within the file itself. The concept of VQ is to represent monochrome or color image blocks with representative kernels from a code book. The indices of the representative kernels replace the image data in the quantized image. The code book and the color lookup table (LUT) are included in the file as overhead information.

2.0 Quantization Process

The VQ algorithm examines each v rows x h columns (v x h) pixel kernel in the input image and uses a clustering technique to develop a limited code book that contains the most representative kernels. The code book entries are v x h pixel kernels. These kernels are interpreted different ways, depending on the type of image they represent. In the case of Red, Green, Blue (RGB/LUT) quantized images, these pixels are actually indices into a color LUT. In other cases, they may represent indices to the gray scale pixel values or spectral band pixel values depending on formats of the gray scale (n-bit), color (RGB/LUT), color bands, R, G, B or multispectral bands. Figure A-3-1 shows the process for vector quantization. The procedure produces the code book and color LUT, if applicable, as part of the VQ header at the beginning of the image data field of the file, as illustrated in Figure A-3-2.

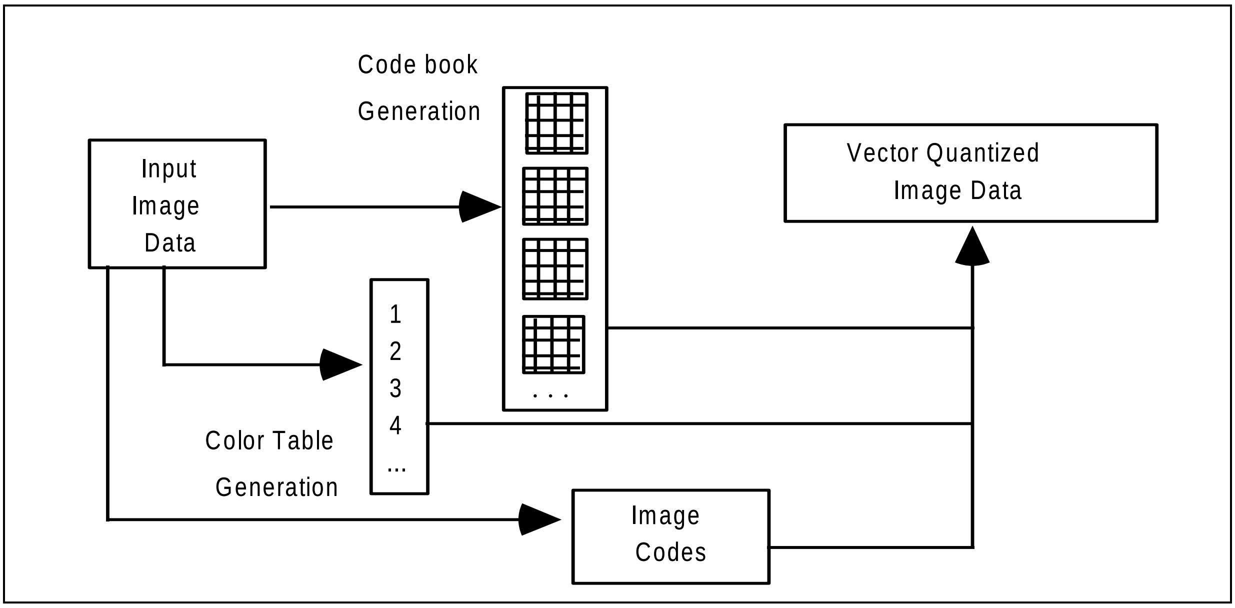

Figure A-3-1 -- Vector Quantization Process Flow

[1] This guidance is taken from ISO/IEC 12087-5:1998(E), Annex B.

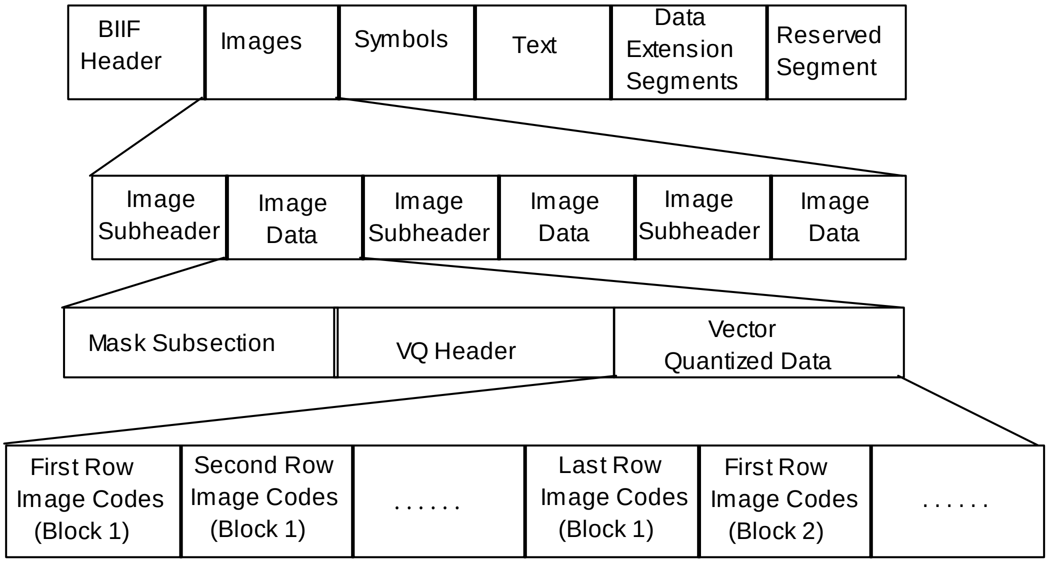

Figure A-3-2 -- NSIF File Structure With VQ Data

3.0 Reconstruction

3.0.1 The VQ data requires only a series of table lookups to reconstruct the image for display. The reconstruction process takes as input the quantized image data, which includes the image codes, code book(s), and color table (when applicable), and by a specified procedure, generates as its output digital reconstructed image data. This standard does not limit the implementation of VQ in terms of the types and sizes of color lookup tables allowed. However, current implementation of VQ within NSIF uses a single RGB/LUT. Other organizations may be implemented in the future.

3.0.2 VQ reconstruction involves replacing image codes in the quantized image with pixel values for use in display or exploitation of the data. If the image has an associated LUT, the reconstruction is performed using the full process, as shown in Figure A-3-3. The image reconstruction is complete at the first step if the quantized image does not have a color LUT. Color reconstruction would not be necessary in cases where the intended output pixel values are placed into the code book. This may occur in the encoding of gray scale imagery or in the encoding of multispectral imagery where each band is quantized separately.

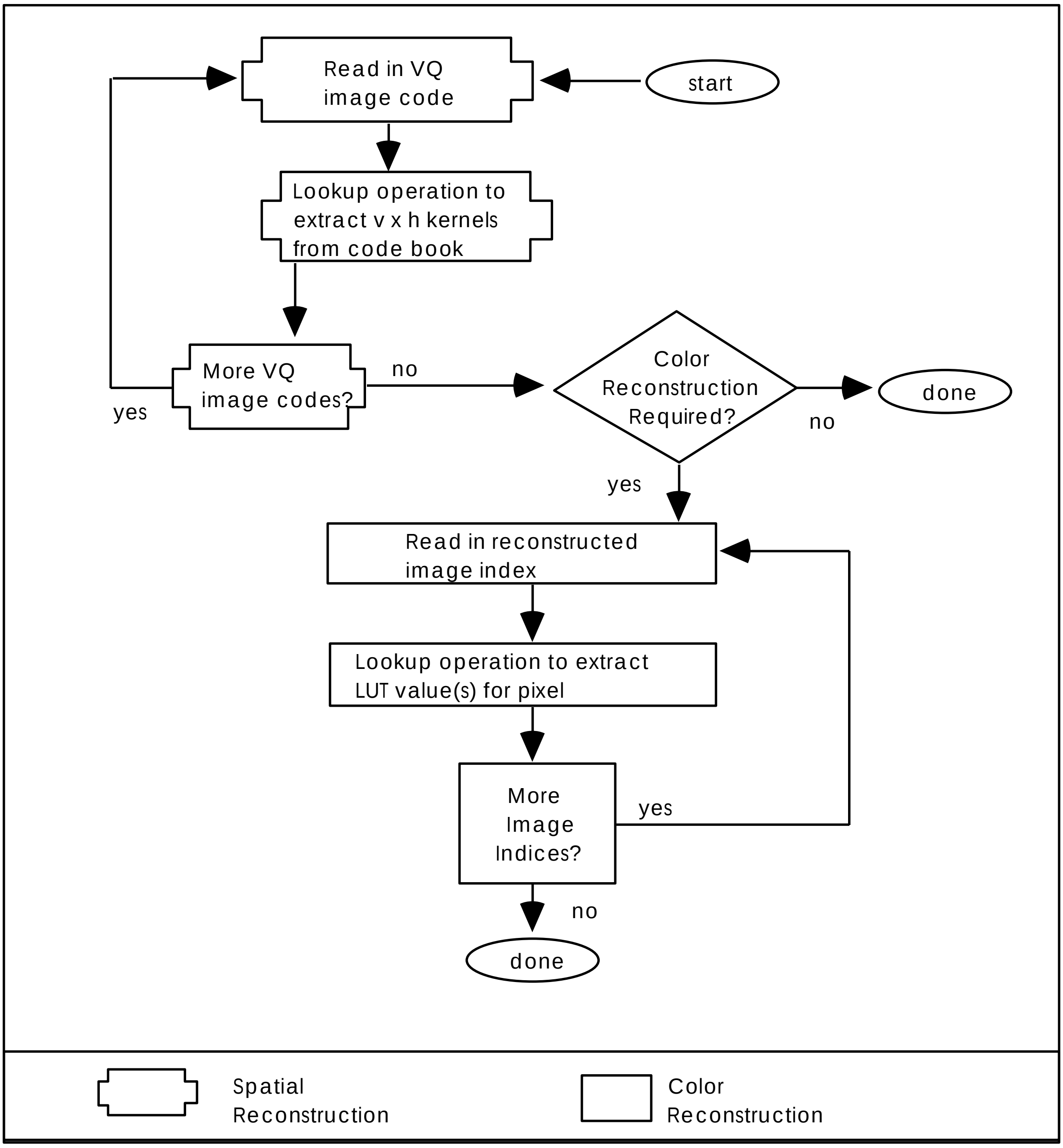

Figure A-3-3 -- VQ Reconstruction Procedure

3.1 Spatial Reconstruction

3.1.1 When the NSIF IC field is set to C4 or M4, the image data field of the VQ formatted NSIF file shall contain a VQ header followed by the quantized image data. The VQ header shall contain information about the data including mask information (M4), and information defining the structure of the compressed image and code book. The code book that is used to reconstruct the image is also contained in the VQ header.

3.1.2 The code book within the image data consists of an array of image codes. Each image code is an index to the code book that has been constructed for the image. Each code book entry logically represents a group of v x h pixel indices. The structure allows for the organization of the VQ code book to be optimized for the specific use of the VQ data. While some NIIA VQ products may require the VQ code book be arranged into v x h index kernels, other products may require that the individual rows for all v x h kernels be stored together such that the image can be reconstructed line-by-line, instead of kernel-by-kernel.

3.1.3 Each of the image codes, during VQ reconstruction, is converted to a kernel (or series of rows) of reconstructed pixel indices. The first image code appearing in the VQ image data field shall be used to spatially reconstruct the v x h indices in the upper left corner of the image. The reconstruction shall continue from left to right across the columns of the first row of image codes, then down each of the rows of image codes sequentially. The output is a spatially decoded image block. If the image has been color quantized, each value in the spatially reconstructed image represents an index into the color table. Figure A-3-4 shows an example of the spatial reconstruction process. Various shades of gray are used to indicate higher or lower values in the code book. If the image is not color quantized, these values would be used to create a gray scale image where higher values in the code book typically correspond to brighter displayed pixels. For a color image, the values in the spatially reconstructed image correspond to indices in the LUT.

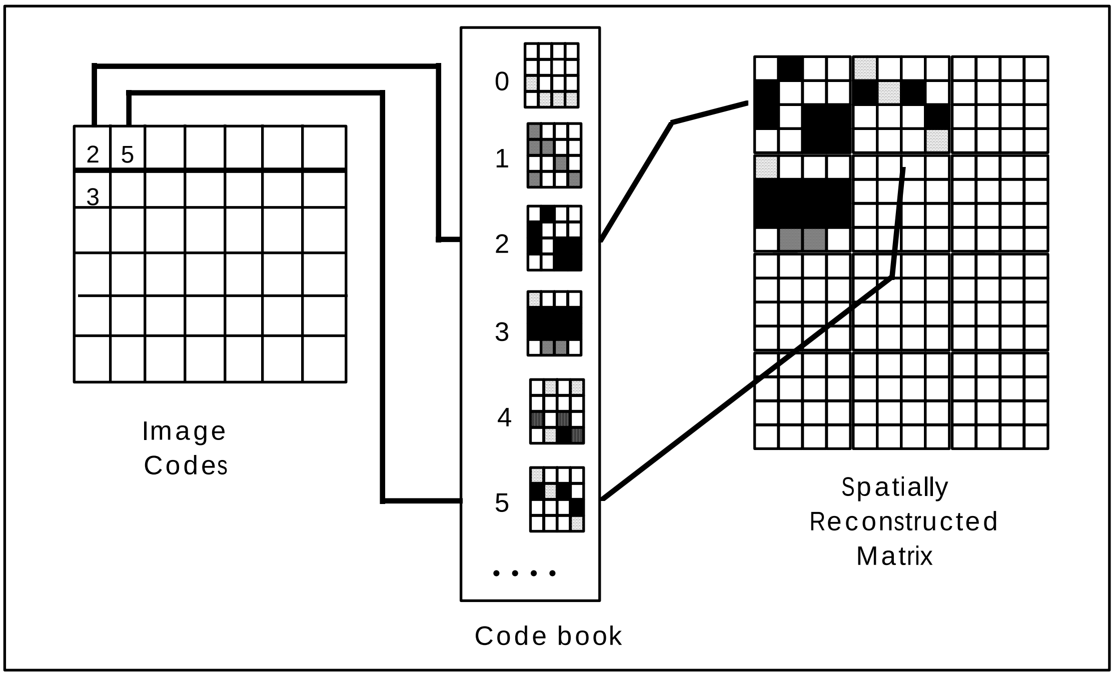

Figure A-3-4 -- Spatial Reconstruction

3.2 Color Reconstruction

Current implementation of VQ within NSIF has a limited scope and uses a single RGB/LUT. The output from the spatial data reconstruction process is an array consisting of values that represent either (1) monochromatic (gray scale) values for an image that is not color coded or (2) indices to the LUT if the image requires the use of a LUT. The final reconstruction step for color quantized images shall transform the indices into the corresponding pixel values by using the LUT values, illustrated in Figure A-3-5.

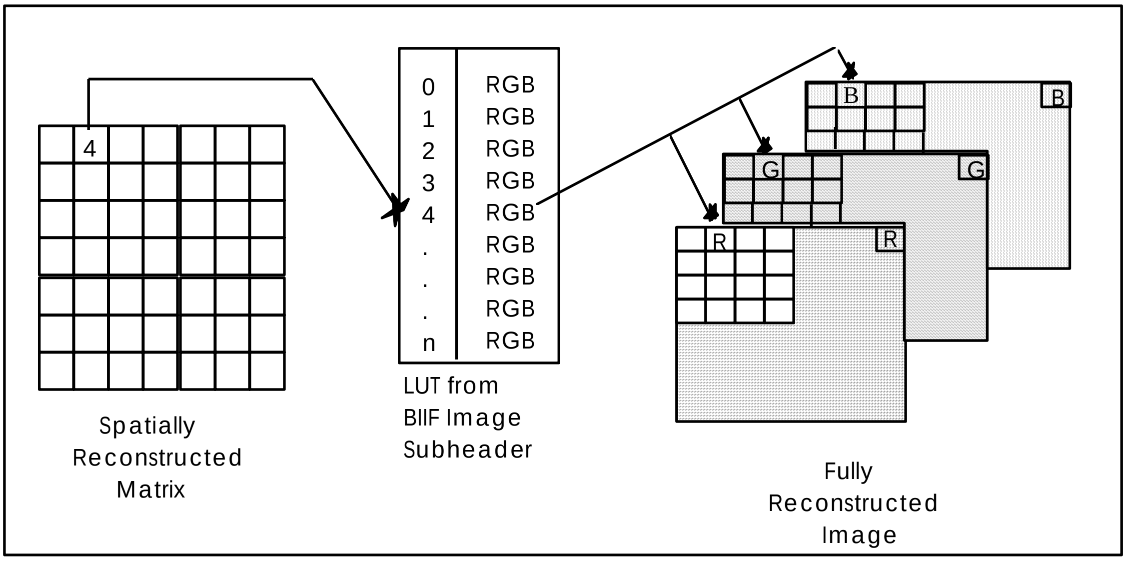

Figure A-3-5 -- Color Reconstruction

3.3 Data Elements for NSIF

The NSIF VQ format allows for many quantization ratios recorded in the COMRAT fields, and several organizations of the code books and color tables. The NSIF VQ file contains the information that the user needs in order to understand the organization of the data and to reconstruct the data for display. The following subclauses describe the fields in the NSIF VQ file that shall be used to determine the VQ organization of a particular file. Further information can also be taken from the vector quantization references provided in the body of this standard. Note: In order to work within the NSIF format structure, vector quantized data elements sometimes use compression/decompression terms in effort to maintain configuration within the NSIF.

3.3.1 Quantization Ratio

Formulae for theoretical and actual vector quantization ratios are provided in the reference documents, and the results are entered in the NSIF Compression Ratio (COMRAT) field of the image subheader. This entry in a VQ file is generalized information and is not used in the reconstruction process. All NSIF VQ files shall contain a value in COMRAT given in the form n.nn representing the average number of bits-per-pixel for the image after vector quantization.

3.3.2 Masked vs Unmasked

For vector quantized images, the Image Compression (IC) field of the image subheader shall contain the value C4 if the image is not masked or M4 if the image is masked. These codes are fully defined in the field definitions of the base NSIF standard.

3.3.3 Code Book Organization

The NSIF VQ image data subclause contains a decompression section where the VQ code book organization is defined. The data includes number of codes in the code book, the size of each v x h kernel, and how the data that make up the kernels are organized. The number of entries in the code book is represented in the <number of decode lookup records> field.

To determine how many pixels make up each kernel, the <number of image rows> field and <number of image codes per row> fields are employed, along with the number of pixels per block vertical (NSIF image subheader field NPPBV) and number of pixels per block horizontal (NSIF image subheader field NPPBH). The following equation is used to determine the size of the kernel in pixels:

$$v = \frac{\text{NPPBV}}{\text{<number of image rows>}} \qquad h = \frac{\text{NPPBH}}{\text{<number of image codes per row>}}$$

kernel size = v rows x h columns

The <number of decode lookup offset records> within the structure shall equal 1 if the data is organized such that all the decode lookup values for each kernel are grouped together.

If the <number of decode lookup offset records> is greater than 1, the data for each kernel is organized into tables. Typically, the tables represent the lookup values for each row of the kernel. The <number of decode lookup records> and the <number of values per decode lookup record> can be used to determine the structure of the code book when the <number of decode lookup tables> is greater than 1.

3.3.4 Spatial Data Section

The spatial data section of the NSIF image data section is organized such that several different file formats (IMODEs), including band interleaved by pixel, band sequential, and band interleaved by block can be accommodated. In addition, the spatial data subsection is partitioned into one or more image block tables (or subframe tables). In all, there are 5 levels of organization above the /image code/ values. Current implementation of VQ within NSIF uses a single band with an associated LUT; therefore IMODE is B, or band interleaved by block.

The [encoded image data] section of the vector quantized file is comprised of:

- the [spectral group] organization, present in the VQ NSIF image data section to allow for the inclusion of multispectral images that are blocked, but are represented as a band sequential image, limited to 1;
- the image, organized into one or more image blocks, each of which is contained in a [subframe table], defined by the number of blocks per row (NBPR) field of the NSIF image subheader and the number of blocks per column (NBPC) field of the NSIF image subheader and identified by the number of [image block tables] in the spatial data subsection;
- one or more [spectral band tables], which define how the pixels are organized;
- the [image row] level of organization corresponding to the <number of image rows> in the VQ header data;
- the [spectral band line] level of organization, corresponding to the <number of image codes per row> in the [image display parameter subheader].

4.0 File Organization

Fields containing identification and origination information, file security information, and the number and size of the data items contained in the NSIF fields are located in the NSIF file header. Information required to decode the file is located in the image subheader and the NSIF VQ image data section. Within the image data section, multi-byte fields are written in the big endian format. Figure A-3-6 is a field-by-field description of the NSIF image data section, as used for a vector quantized file. The mask subsection (shown in schematic on the Figure A-3-2) is shown at a high level only. The specific fields and definitions for the mask subsection are provided in the NSIF base standard.

```
[NSIF image data]
        {2}
        <blocked image data offset>,uint:4(0,1)
        [mask subsection] (0,1)
                {3}
                [mask subheader]
                [block mask table] (0,1)
                [pad pixel mask table] (0,1)
        {2}
        [VQ Header]
                {3}
                [image display parameter sub-header]
                        {4}
                        <number of image rows>, uint:4
                        <number of image codes per row>,uint:4
                        <image code bit length>,uint:1
                {3}
                [decode section] (0,1)
                        {4}
                        [decode section subheader]
                                {5}
                                <decode algorithm id>,uint:2
                                <number of decode lookup offset records>, uint:2
                                <number of decode parameter offset records>,uint:2
                                        {6}
                                        [decode lookup offset record] (1, ....many)
                        {4}
                        [decode lookup subsection] (0,1)
                                {5}
                                <decode lookup offset table offset>, uint:2
                                <decode lookup table offset record length>, uint:2
                                [decode offset lookup table]
                                                {7}
                                                <decode lookup table id>, uint:2
                                                <number of decode lookup records>, uint:4
                                                <number of values per decode lookup record>,uint:2
                                                <decode lookup value bit length>, uint:2
                                                <decode lookup table offset>,uint:4
                                {5}
                                [decode lookup table] (1, ...many)
                                        {6}
                                        [decode lookup record] (1,... many)
                                                {7}
                                                /decode lookup value/, bits:var (1,  ... many)
        {2}
        [encoded image data]
                {3}
                [spectral group] (1, ...many)
                        {4}
                        [subframe table] (1, ....many)
                                {5}
                                [spectral band table] (1, ....many)
                                        {6}
                                        [image row] (1, ... many)
                                                {7}
                                                [spectral band line] (1, ...many)
                                                        {8}
                                                        /image code/,bits:var(1, ... many)
```

Figure A-3-6 -- Structure of the NSIF VQ Image Data Section

5.0 Definitions - Image Data Section

This subclause lists the elements of the VQ header and quantized image data sections, listed in alphabetical order.

1. (<blocked image data offset> ::= a 4-byte unsigned integer defining the offset in bytes of the [encoded image data] from the beginning of the section (labeled "image data" in Figure B.2). This field is present only for masked images.
2. <decode algorithm id> ::= a 2-byte unsigned integer defining the algorithm used for the image data in the [frame file]. ::= 1 to indicate that this image data is vector quantized.
3. <decode lookup offset table offset> ::= a 4-byte unsigned integer indicating the displacement, measure in bytes, between the beginning of the [decode lookup subsection] and the first byte of the decode lookup offset table]. The first byte of the [decode lookup subsection] is counted as 0.
4. <decode lookup table id> ::= a 2-byte unsigned integer identifying the [lookup table] described in this [decode lookup offset record], encoded as follows
    a) = 1 to indicate this is row 0 of a 4 x 4 kernel,
    b) = 2 to indicate this is row 1 of a 4 x 4 kernel,
    c) = 3 to indicate this is row 2 of a 4 x 4 kernel,
    d) = 4 to indicate this is row 3 of a 4 x 4 kernel,
    e) = 5 to indicate this is a 16-element, 4 x 4 kernel,
    f) = 6 to indicate this is a 4-element, 2 x 2 kernel.

    The nth[decode lookup offset record] shall contain the <decode lookup table id> of the nth [decode lookup table] in this [decode lookup subsection].
5. <decode lookup table offset> ::= a 4-byte unsigned integer defining the displacement, measured in bytes, between the beginning of the [decode lookup subsection] and the first byte of the [decode lookup table] identified in this [decode lookup offset record]. The first byte of the [decode lookup subsection] is counted as 0.
6. <decode lookup table offset record length> ::= a 2-byte unsigned integer indicating the length of each [decode lookup offset record].
7. <decode lookup value> ::= a variable-length bit field specifying a value in the VQ code book. For a particular VQ scheme, the [decode lookup value] shall have affixed length, which is defined in the <decode lookup value bit length>.
8. <decode lookup value bit length> ::= a 2-byte unsigned integer >= 4, defining the length in bits of the /decode lookup value/ field in each [decode lookup record] of each [decode lookup table] in the [decode section]. All [decode lookup value] fields in a given [decode lookup table] shall have the same <decode lookup value bit length>, which shall be a multiple of 4 bits.
9. <image code> ::= a variable-length bit string indicating an index to the associated VQ code book in a vector quantized map or image file. Successive [image code] values in a given [image row] shall be a multiple of 8 bits, to ensure that each [image row] consists of an integer number of bytes.

10. <image code bit length> ::= a 1-byte unsigned integer defining the length, in bit, of /image code/.
11. <number of decode lookup records> ::= a 4-byte unsigned integer >= 1, indicating the number of [decode lookup record]s in each [decode lookup table].
12. <number of decode lookup offset records> ::= a 2-byte unsigned integer ≥ 1, indicating the number of [decode lookup offset record]s in the [decode lookup offset table].
13. <number of decode parameter offset records> ::= a 2-byte unsigned integer >=0, indicating the number of [decode parameter offset record]s in the [decode parameter subsection]. For VQ images, no [decode parameter offset record] is present and therefore, this value shall ::= 0.
14. <number of image codes per row> ::= a 4-byte unsigned integer >= 1, defining, the number of [image code] fields in each [image row] of each [color band table]. All [image row]s in every [spectral band table] in every [subframe table] shall contain the same number of contiguous [image codes]. The <number of image codes per row> shall be chosen to ensure that the total number of bits in the/image code/s constituting a give [image row] shall be a multiple of 8 bits, to ensure that each [image row] consists of an integer number of bytes.
15. <number of image rows> ::= a 4-byte unsigned integer >= 1, indicating the number of [image row]s in each [spectral band table]. All [spectral band table]s in every [subframe table] shall contain the same number of [image row]s.
16. <number of values per decode lookup record> ::= a 2-byte unsigned integer >= 1, indicating the number of contiguous [decode lookup value] fields in each [decode lookup record] of a given [decode lookup table]. All [decode lookup table]s in a given [decode section] shall have the same number of [decode lookup value] fields in each [decode lookup record].

6.0 Definitions - NSIF Header and Image Subheader

Table A-3-1 provides specific data values for NSIF header and image subheader fields particular to VQ data.

Table A-3-1 -- NSIF Header and Subheader Specified Data Values

FIELD	NAME	SIZE	VALUE RANGE	TYPE
IC	Image Compression. Specific values are identified for masked and unmasked vector quantized images.	2	For VQ images: C4 = VQ image, not masked M4 = VQ image, masked	R
PVTYPE	Pixel Value Type. Type of computer representation used for the value of each /image code/ in the NSIF image.	3	For VQ images: INT = integer	R
CLEVEL	Complexity Level. This field shall contain the complexity level required to interpret fully all components of the file.	2	Valid entries are integers assigned in accordance with complexity requirements established in ANNEX C of this Appendix.	R

Annex A, Appendix 4
Lossless JPEG Implementation Considerations[1]

1.0 Introduction

This standard is intended to enable the interchange of 2 to 16 bit gray scale imagery and 24 bit color imagery. ISO/IEC 10918-1 represents a collection of lossy and lossless compression techniques, a subset of the lossless procedures are used in generation of the compressed image data stream included in this Appendix. Unless expressly forbidden in this document, any procedure in ISO/IEC 10918-1 applicable to lossless encoding may be applied. Any optional processes in ISO/IEC 10918-1 required by this Appendix will be detailed.

1.1 Encoders

Encoders shall output a full interchange format that includes the compressed image data and all table specifications used in the encoding process.

1.2 Decoders

All decoders shall interpret full interchange format. Abbreviated interchange format decoders are not a requirement of this standard.

1.3 Markers and Tags

The following tables specify the markers and tag usage from ISO/IEC 10918-1 and ISO/IEC 10918-3 applicable to the Lossless JPEG profile.

Table A-4-1 - Marker Usage
(ISO/IEC 10918-1, JPEG part 1)

Symbol	*Description*	*Parameters*	*Req.*	*Cap.*	*Exc.*
Start Of Frame markers, non-differential, Huffman coding					
SOF_0	Baseline DCT	Table 4			X
SOF_1	Extended sequential DCT				X
SOF_2	Progressive DCT				X
SOF_3	Lossless (sequential)		X		
Start Of Frame markers, differential, Huffman coding					
SOF_5	Differential sequential DCT				X
SOF_6	Differential progressive DCT				X
SOF_7	Differential lossless (sequential)				X
Start Of Frame markers, non-differential, arithmetic coding					
SOF_9	Extended sequential DCT				X
SOF_{10}	Progressive DCT				X
SOF_{11}	Lossless (sequential)				X
Start Of Frame markers, differential, arithmetic coding					
SOF_{13}	Differential sequential DCT				X
SOF_{14}	Differential progressive DCT				X
SOF_{15}	Differential lossless (sequential)				X
Huffman table specification					

[1] This guidance is taken from U.S. NIMA Document N-0106-97.

Table A-4-1 - Marker Usage (ISO/IEC 10918-1, JPEG part 1)					
Symbol	*Description*	*Parameters*	*Req.*	*Cap.*	*Exc.*
DHT	Define Huffman table(s)	Table 5		X	
Arithmetic coding conditioning specification					
DAC	Define arithmetic coding conditioning(s)	Table 6			X
Restart interval termination					
RSTm	Restart with module 8 count “m”		X		
Other markers					
Marker Usage (ISO/IEC 10918-1, JPEG part 2)					
Symbol	*Description*	*Parameters*	*Req.*	*Cap.*	*Exc.*
SOI	Start of image		X		
EOI	End of image		X		
SOS	Start of scan	Table 7	X		
DQT	Define quantization table(s)	Table 8			X
DNL	Define number of lines	Table 12			X
DRI	Define restart interval	Table 9	X		
DHP	Define hierarchical progression	see ISO/IEC 10918-1			X
EXP	Expand reference component(s)	Table 13			X
APPn	Reserved for application segments	Table 11		X	
COM	Comment	Table 10		X	

Table A-4-2 - Marker Usage (ISO/IEC 10918-3, JPEG part 3)					
Symbol	*Description*	*Parameters*	*Req.*	*Cap.*	*Exc.*
Version 1 Extensions					
VER	Version	Table 15			X
DTI	Define tiled image	Table 20			X
DTT	Define tile	Table 21			X
SRF	Selectively refined frame	Table 18			X
SRS	Selectively refined scan	Table 19			X
DCR	Define component registration	Table 22			X
DQS	Define quantizer scale selection	Table 23			X

Table A-4-3 - SPIFF Tags Usage (ISO/IEC 10918-3, JPEG part 3 cont.)				
SPIFF Tags	*Parameters*	*Req.*	*Cap.*	*Exc.*
SPIFF header	Table 14			X
Transfer characteristics	Table 24			X
Component registration	Table 25			X
Image orientation	Table 26			X
Thumbnail	Table 27			X
Image title	Table 28			X
Image description	Table 29			X
Time stamp	Table 30			X
Version identifier	Table 31			X
Creator identification	Table 32			X
Protection indicator	Table 33			X
Copyright information	Table 34			X
Contact information	Table 35			X

Table A-4-3 - SPIFF Tags Usage (ISO/IEC 10918-3, JPEG part 3 cont.)				
SPIFF Tags	*Parameters*	*Req.*	*Cap.*	*Exc.*
Tile index	Table 36			X
Scan index	Table 37			X
Set reference	Table 38			X

1.4 Marker and Tag Parameterization

The following tables specify the values and range of values allowed for all required and capable markers. For clarity, whenever a parameterization is between one of a few choices that significantly alters a table's size or structure, multiple versions of that table are included, one for each parameterization. If a given table is not applicable to this standard, it will be indicated by "N/A" in its parameter specifications. This standard provides for the lossless encoding of 2-16 bit gray scale and 24 bit RGB color imagery. Many tables therefore have two parameterizations depending on imagery type. In the following tables, the parameter specifications for gray scale imagery are given first, followed by those for RGB color imagery. The parameters associated with a given image type cannot be mixed with those of another image type. For example, if we are using RGB imagery, then in Table A-4-4, Lf = 17 and P = 8. These are the only allowed combination of parameters. If only one parameter specification is given for any parameter in a table, it applies to both image types.

Table A-4-4 - Frame Header (SOF) (See ISO/IEC 10918-1 Table C.2)						
		Values				
Parameter	*Size (bits)*	*Sequential DCT*		*Progressive DCT*	*Lossless*	*Profile Parameter Specifications (Gray, RGB)*
		Baseline	*Extended*			
Lf	16	8 + 3 ´ Nf				11, 17
P	18	8-255	8, 12	8, 12	2-165	2-16, 8
Y	16	$0 \leq Y \leq 2^{16} – 1$				$1 \leq Y \leq 2^{16} - 1$
X	16	$1 \leq X \leq 2^{16} – 1$				$1 \leq X \leq 2^{16} - 1$
Nf	18	1-255	1-255	1-4	1-255	1, 3
C_i	18	0-25535				0, 0-2
H_i	14	1-43550				1
V_i	14	1-43550				1
T_{qi}	18	0-312	0-355	0-3	0-125	0

Table A-4-5 - Huffman Table Specification (DHT) (See ISO/IEC 10918-1 Table C.5)						
		Values				
Parameter	*Size (bits)*	*Sequential DCT*		*Progressive DCT*	*Lossless*	*Profile Parameter Specifications (Gray, RGB)*
		Baseline	*Extended*			

Lh	16					22-36, [28, 29, 54, 56, 80, 83] see Table 40
Tc	4	0,1			0	0
Th	4	0,1	0-3			0, 0-2
	8	0-255				0-255
$V_{i,j}$	8	0-255				0-255

Table A-4-6 - Arithmetic Coding Conditioning Table-Specification (DAC) (See ISO/IEC 10918-1 Table C.6)						
		Values				
Parameter	*Size (bits)*	*Sequential DCT*		*Progressive DCT*	*Lossless*	*Profile Parameter Specifications*
		Baseline	*Extended*			
La	16	Undefined	2 + 2 x n			N/A
Tc	4	Undefined	0,1		0	N/A
Tb	4	Undefined	0-3			N/A
Cs	8	Undefined	0-255 (Tc=0), 1-63 (Tc=1)		0-255	N/A

Table A-4-7 - Scan Header (SOS) (See ISO/IEC 10918-1 Table C.3)						
		Values				
Parameter	*Size (bits)*	*Sequential DCT*		*Progressive DCT*	*Lossless*	*Profile Parameter Specifications (Gray, RGB)*
		Baseline	*Extended*			
Ls	16	6 + 2 ´ Ns				8, 12
Ns	18	1-4				1, 3
Csj	18	0-255 [a]				0, 0-2
Tdj	14	0-1	0-3	0-3	0-3	0, 0-2
Taj	14	0-1	0-3	0-3	0	0
Ss	18	0-1	0-1	0-63	1-7 [b]	1-7
Se	18	63-	63-	Ss-63 [c]	0	0
Ah	14	0-1	0-1	0-13	0	0
Al	14	0-1	0-1	0-13	0-15	0-15
a) Csj shall be a member of the set of Ci specified in the frame header. b) 0 for lossless differential frames in the hierarchical mode (see C.3 of ISO/IEC 10918-1) c) 0 if Ss equals zero.						

Table A-4-8 - Quantization Table Specification (DQT) (See ISO/IEC 10918-1 Table C.4)						
		Values				
Parameter	*Size (bits)*	*Sequential DCT*		*Progressive DCT*	*Lossless*	*Profile Parameter Specifications*
		Baseline	*Extended*			
Lq	16	$2 + \sum_{t=1}^{n}(65 + 64 \times Pq(t))$			Undefined	N/A
Pq	4	0	0,1	0,1	Undefined	N/A
Tq	4	0-3			Undefined	N/A
Qk	8,16	1-255, $1 \le Qk \le 2^{16} - 1$			Undefined	N/A

Table A-4-9 - Define Restart Interval Segment (DRI) (See ISO/IEC 10918-1 Table C.7)						
		Values				
Parameter	*Size (bits)*	*Sequential DCT*		*Progressive DCT*	*Lossless*	*Profile Parameter Specifications*
		Baseline	*Extended*			
Lr	16	4				4
Ri	16	$0 \le Ri \le 2^{16} - 1$			n x MCUR	1-8

Table A-4-10 - Comment Segment (COM) (See ISO/IEC 10918-1 Table C.8)						
		Values				
Parameter	*Size (bits)*	*Sequential DCT*		*Progressive DCT*	*Lossless*	*Profile Parameter Specifications*
		Baseline	*Extended*			
Lc	16	$2 \le Lc \le 2^{16}–1$				$2 \le Lc \le 2^{16}-1$
Cm_i	8	0-255				0-255

Table A-4-11 - Application Data Segment (APPn) (See ISO/IEC 10918-1 Table C.9)						
		Values				
Parameter	*Size (bits)*	*Sequential DCT*		*Progressive DCT*	*Lossless*	*Profile Parameter Specifications*
		Baseline	*Extended*			
Lp	16	$2 \le Lp \le 2^{16}–1$				$2 \le Lp \le 2^{16}-1$
Api	18	0-25522				0-25522

Table A-4-12 - Define Number of Lines Segment (DNL) (See ISO/IEC 10918-1 Table C.10)						
		Values				
Parameter	*Size (bits)*	*Sequential DCT*		*Progressive DCT*	*Lossless*	*Profile Parameter Specifications*
		Baseline	*Extended*			
Ld	16	4-65535 a				N/A
NL	16	$1 \le NL \le 2^{16} - 1$ [a]				N/A
a) The value specified shall be consistent with the number of lines coded at the point where the DNL segment terminates the compressed data segment.						

Table A-4-13 - Expand Segment (EXP) (See ISO/IEC 10918-1 Table C.11)					
		Values			
Parameter	*Size (bits)*	*Sequential DCT* *Extended*	*Progressive DCT*	*Lossless*	*Profile Parameter Specifications*
Le	16	3			N/A
Eh	14	0, 1			N/A
Ev	14	0, 1			N/A

The remaining tables of this section deal with extensions and file formats in ISO/IEC 10918-3. This Appendix does not make use of these features and they are therefore not applicable. No markers or tags associated with ISO/IEC 10918-3 will appear in a file or data stream compliant to this profile.

Table A-4-14 - SPIFF File Header (See ISO/IEC 10918-3 Table F.1)			
Parameter	*Type, Size*	*Values*	*Profile Parameter Specifications*
MN	I.32	X' FFD8FFE8'	N/A
HLEN	I.16	32	N/A
IDENT	S.6	X' 535049464600'	N/A
VERS	I.16	X' 0100'	N/A
P	I.8	0 – 4	N/A
NC	I.8	1 – 255	N/A
HEIGHT	I.32	$1 \leq$ HEIGHT $\leq 2^{32} - 1$	N/A
WIDTH	I.32	$1 \leq$ WIDTH $\leq 2^{32} - 1$	N/A
S	I.8	0 – 15	N/A
BPS	I.8	1,2,4,8,12,16	N/A
C	I.8	0 – 5	N/A
R	I.8	0 – 2	N/A
VRES	F / I.32	$1 \leq$ VRES $\leq 2^{32} - 1$	N/A
HRES	F / I.32	$1 \leq$ HRES $\leq 2^{32} - 1$	N/A

Table A-4-15 - Version Marker Segment (VER) (See ISO/IEC 10918-3 Table C.3)						
		Values				
Parameter	*Size (bits)*	*Sequential DCT*		*Progressive DCT*	*Lossless*	*Profile Parameter Specifications*
		Baseline	*Extended*			
Lv	16	5, V = 0 6, V = 1				N/A
V	8	0, 1				N/A
Rev	8	0				N/A
CAPi	8 8	CAP0, version 0, see Table 16 CAP1, version 1, see Table 17				N/A

Table A-4-16 - Capability Indicator Byte for Version 0 (JPEG part 1)				
Coding Process (ISO/IEC 10918-1)	*CAP 0 Value*	*Req.*	*Cap.*	*Exc.*
Baseline sequential	0000 0000	N/A	N/A	N/A
Extended sequential, Huffman, 8-bits	0000 0001	N/A	N/A	N/A
Extended sequential arithmetic, 8-bits	0000 0011	N/A	N/A	N/A
Extended sequential Huffman, 12-bits	0000 0101	N/A	N/A	N/A
Extended sequential arithmetic, 12-bits	0000 0111	N/A	N/A	N/A
Spectral selection Huffman, 8-bits	0001 0001	N/A	N/A	N/A
Spectral selection arithmetic, 8-bits	0001 0011	N/A	N/A	N/A
Full progression Huffman, 8-bits	0001 1001	N/A	N/A	N/A
Full progression arithmetic, 8-bits	0001 1011	NA/	N/A	N/A
Spectral selection Huffman, 12-bits	0001 0101	N/A	N/A	N/A
Spectral selection arithmetic, 12-bits	0001 0111	N/A	N/A	N/A
Full progression Huffman, 12-bits	0001 1101	N/A	N/A	N/A
Full progression arithmetic, 12-bits	0001 1111	N/A	N/A	N/A
Lossless Huffman	0010 0001	N/A	N/A	N/A
Lossless arithmetic	0010 0011	N/A	N/A	N/A
Hierarchical, sequential Huffman, 8-bits	0100 0001	N/A	N/A	N/A
Hierarchical, sequential arithmetic, 8-bits	0100 0011	N/A	N/A	N/A
Hierarchical, sequential Huffman, 12-bits	0100 0101	N/A	N/A	N/A
Hierarchical, sequential arithmetic, 12-bits	0100 0111	N/A	N/A	N/A
Hierarchical, Spectral selection Huffman, 8-bits	0101 0001	N/A	N/A	N/A
Hierarchical, Spectral selection arithmetic, 8-bits	0101 0011	N/A	N/A	N/A
Hierarchical, Full progression Huffman, 8-bits	0101 1001	N/A	N/A	N/A
Hierarchical, Full progression arithmetic, 8-bits	0101 1011	N/A	N/A	N/A
Hierarchical, Spectral selection Huffman, 12-bits	0101 0101	N/A	N/A	N/A
Hierarchical, Spectral selection arithmetic, 12-bits	0101 0111	N/A	N/A	N/A
Hierarchical, Full progression Huffman, 12-bits	0101 1101	N/A	N/A	N/A
Hierarchical, Full progression arithmetic, 12-bits	0101 1111	N/A	N/A	N/A
Hierarchical, Lossless Huffman	0110 0001	N/A	N/A	N/A
Hierarchical, Lossless arithmetic	0110 0011	N/A	N/A	N/A

Table A-4-17 - Capability Indicator Byte for Version 1 (JPEG part 3)				
Capability (ISO/IEC 10918-3)	*Bit positions*	*Req.*	*Cap.*	*Exc.*
10 < blocks per MCU ≤ 20	0xxx xxx1	N/A	N/A	N/A
Variable quantization	0xxx xx1x	N/A	N/A	N/A
Hierarchical selective refinement	0xxx x1xx	N/A	N/A	N/A
Progressive selective refinement	0xxx 1xxx	N/A	N/A	N/A
Component selective refinement	0xx1 xxxx	N/A	N/A	N/A
Simple tiling	001x xxxx	N/A	N/A	N/A
Pyramidal tiling	010x xxxx	N/A	N/A	N/A
Composite tiling	011x xxxx	N/A	N/A	N/A
Note - ‘ x’ indicates ‘ don’ t care’				

Table A-4-18 - Selectively Refined Frame (SRF) (See ISO/IEC 10918-3 Table C.6)			
Parameter	*Size (bits)*	*Values*	*Profile Parameter Specifications*
Lrf	16	6	N/A
Ovf	16	$0 \le Ovf \le 2^{16} - 1$	N/A
Ohf	16	$0 \le Ohf \le 2^{16} - 1$	N/A

Table A-4-19 - Selectively Refined Scan (SRS) (See ISO/IEC 10918-3 Table C.7)			
Parameter	*Size (bits)*	*Values*	*Profile Parameter Specifications*
Lrs	16	10	N/A
Ovs	16	$0 \le Ovs \le 2^{16} - 1$	N/A
Ohs	16	$0 \le Ohs \le 2^{16} - 1$	N/A
Svs	16	$1 \le Svs \le 2^{16} - 1$	N/A
Shs	16	$1 \le Shs \le 2^{16} - 1$	N/A

Table A-4-20 - Define Tiled Image (DTI) (See ISO/IEC 10918-3 Table C.8)			
Parameter	*Size (bits)*	*Values*	*Profile Parameter Specifications*
Lti	16	15	N/A
TT	8	0 = simple, 1 = pyramidal, 2 = composite	N/A
Tivs	16	1 for simple and pyramidal tiling $1 \le TIvs \le 2^{16}$ -1 for composite tiling	N/A
This	16	1 for simple and pyramidal tiling $1 \le TIhs \le 2^{16}$ -1 for composite tiling	N/A
RGvs	32	$1 \le RGvs \le 2^{32} - 1$	N/A
RGhs	32	$1 \le RGhs \le 2^{32} - 1$	N/A

Table A-4-21 - Define tile (DTT) (See ISO/IEC 10918-3 Table C.9)			
Parameter	*Size (bits)*	*Values*	*Profile Parameter Specifications*
Ltf	16	18	N/A
TFvs	32	$1 \le TFvs \le 2^{32} - 1$	N/A
TFhs	32	$1 \le TFhs \le 2^{32} - 1$	N/A
Tfvo	32	$0 \le TFvo \le 2^{32} - 1$	N/A
Tfho	32	$0 \le TFho \le 2^{32} - 1$	N/A

Table A-4-22 - Define Component Registration (DCR) (See ISO/IEC 10918-3 Table C.10)			
Parameter	*Size (bits)*	*Values*	*Profile Parameter Specifications*
Lcr	16	4	N/A
Ci	8	0 ≤ Ci ≤ 255	N/A
Crvo	4	0 ≤ CRvo ≤ 8	N/A
Crho	4	0 ≤ CRho ≤ 8	N/A

Table A-4-23 - Define Quantizer Scale Selection (DQS) (See ISO/IEC 10918-3 Table C.11)			
Parameter	*Size (bits)*	*Values*	*Profile Parameter Specifications*
Lqs	16	3	N/A
Tc	8	Tc = 0 indicates a linear table Tc = 1 indicates a non-linear table	N/A

Table A-4-24 - Transfer Characteristics (See ISO/IEC 10918-3 Table F.6)				
Transfer characteristics			*Tag value: X' 00000002'*	*Profile Parameter Specifications*
Offset	*Parameter*	*type, size*	*Values*	
0	TRANCHAR	I.8	1-8	N/A
1	RESERVED	C.3	0	N/A

Table A-4-25 - Component Registration (See ISO/IEC 10918-3 Table F.7)				
Component registration			*Tag value: X' 00000003'*	*Profile Parameter Specifications*
Offset	*Parameter*	*type, size*	*Values*	
0	$CROFFSET_0$	I.8	0 – 255	N/A
1	$CROFFSET_1$	I.8	0 – 255	N/A
2	**...**			

Table A-4-26 - Image Orientation (See ISO/IEC 10918-3 Table F.8)				
Image orientation			*Tag value: X' 00000004'*	*Profile Parameter Specifications*
Offset	*Parameter*	*type, size*	*Values*	
0	IMGOR	I.8	0 – 3	N/A
1	IMGFLIP	I.8	0, 1	N/A
2	RESERVED	C.2	0	N/A

Table A-4-27 - Thumbnail Image Specification (See ISO/IEC 10918-3 Table F.9)				
Thumbnail image specification			*Tag value: X' 00000005'*	*Profile Parameter Specifications*
Offset	*Parameter*	*type, size*	*Values*	
0	TNDATA	I.32	Any	N/A
4	TNHEIGHT	I.16	$1 \leq$ TNHEIGHT $\leq 2^{16} - 1$	N/A
6	TNWIDTH	I.16	$1 \leq$ TNWIDTH $\leq 2^{16} -1$	N/A
8	TNS	I.8	0 – 14	N/A
9	TNBPS	I.8	1,2,4,8,12,16	N/A
10	TNC	I.8	0 – 5	N/A
11	RESERVED	C.1	0	N/A
12	...			

Table A-4-28 - Image Title (See ISO/IEC 10918-3 Table F.10)				
Image Title			*Tag value: X' 00000006'*	*Profile Parameter Specifications*
Offset	*Parameter*	*type, size*	*Values*	
0	TITLELOC	I.32	0 or in range from EOI marker offset to $2^{32} – 1$	N/A
4	CHARSET	I.8	1 to N, where N is largest existing ISO/IEC 8859-N, 254, 255	N/A

Table A-4-29 - Image Description (See ISO/IEC 10918-3 Table F.11)				
Image description			*Tag value: X' 00000007'*	*Profile Parameter Specifications*
Offset	*Parameter*	*type, size*	*Values*	
0	DESCLOC	I.32	0 or in range from EOI marker offset to $2^{32} – 1$	N/A
4	CHARSET	I.8	1 to N, where N is largest existing ISO/IEC 8859-N, 254, 255	N/A

Table A-4-30 - Time Stamp (See ISO/IEC 10918-3 Table F.12)				
Time Stamp			*Tag value: X' 00000008'*	*Profile Parameter Specifications*
Offset	*Parameter*	*type, size*	*Values*	
0	DATE	S.10	ISO 8601 format date	N/A
10	TIME	S.13	ISO 8601 format time	N/A
23	RESERVED	C.1	0 (reserved)	

Table A-4-31 - Version Identifier (See ISO/IEC 10918-3 Table F.13)				
Version identifier			*Tag value: X' 00000009'*	*Profile Parameter Specifications*
Offset	*Parameter*	*type, size*	*Values*	
0	VERSNLOC	I.32	0 or in range from EOI marker offset to $2^{32} – 1$	N/A
4	CHARSET	I.8	1 to N, where N is largest existing ISO/IEC 8859-N, 254, 255	N/A
5	**...**			

Table A-4-32 - Creator Identification (See ISO/IEC 10918-3 Table F.14)				
Creator Identification			*Tag value: X' 0000000A'*	*Profile Parameter Specifications*
Offset	*Parameter*	*type, size*	*Values*	
0	CREATLOC	I.32	0 or in range from EOI marker offset to 2 32 –1	N/A
4	CHARSET	I.8	1 to N, where N is largest existing ISO/IEC 8859-N, 254, 255	N/A
5	**...**			

TableA-4-33 - Protection Indicator (See ISO/IEC 10918-3 Table F.15)				
Protection Indicator			*Tag value: X' 0000000B'*	*Profile Parameter Specifications*
Offset	*Parameter*	*type, size*	*Values*	
0	LEVAUT	I.8	0-3	N/A
1	COPYRID	I.8	0-255	N/A
2	RESERVED	C.2	0 (reserved)	

Table A-4-34 - Copyright Information (See ISO/IEC 10918-3 Table F.16)				
Copyright Information			*Tag value: X' 0000000C'*	*Profile Parameter Specifications*
Offset	*Parameter*	*type, size*	*Values*	
0	COPYRLOC	I.32	0 or in range from EOI marker offset to 2 32 – 1	N/A
4	CHARSET	I.8	1 to N, where N is largest existing ISO/IEC 8859-N, 254, 255	N/A

Table A-4-35 - Contact Information (See ISO/IEC 10918-3 Table F.17)				
Contact Information			*Tag value:* *X' 0000000D'*	*Profile Parameter Specifications*
Offset	*Parameter*	*type, size*	*Values*	
0	REGCON	I.16	$1 \leq$ REGCON $\leq 2^{16} - 1$, interpreted as ISO 3166 numeric country code. A value of X' 0000' indicates an international organization.	N/A
2	REGAUT	I.16	$0 \leq$ REGAUT $\leq 2^{16} – 1$	N/A
4	REGID	I.32	$0 \leq$ REGID $\leq 2^{32} – 1$	N/A
8	CONTLOC	I.32	0 or in range from EOI marker offset to $2^{32} – 1$	N/A
12	CHARSET	I.8	1 to N, where N is largest existing ISO/IEC 8859-N, 254, 255	N/A
13	**...**			

Table A-4-36 - Tile Index (See ISO/IEC 10918-3 Table F.18)				
Tile Index			*Tag value:* *X' 0000000E'*	*Profile Parameter Specifications*
Offset	*Parameter*	*type, size*	*Values*	
0	DTTINDX	I.32	0 or in range from EOI marker offset to $2^{32} – 1$	N/A
4	NUMDTT	I.32	$0 \leq$ NUMDTT $\leq 2^{32} – 1$	N/A

Table A-4-37 - Scan Index (See ISO/IEC 10918-3 Table F.19)				
Scan Index			*Tag value:* *X' 0000000F'*	*Profile Parameter Specifications*
Offset	*Parameter*	*type, size*	*Values*	
0	SCANLIST	I.32	0 or in range from EOI marker offset to $2^{32} – 1$	N/A
4	NUMSCAN	I.32	$0 \leq$ NUMSCAN $\leq 2^{32} – 1$	N/A

Table A-4-38 - Set Reference (See ISO/IEC 10918-3 Table F.20)				
Set Reference			*Tag value:* *X' 00000010'*	*Profile Parameter Specifications*
Offset	*Parameter*	*type, size*	*Values*	
0	REFNO1	I.32	$0 \leq$ REFNO1 $\leq 2^{32} – 1$	N/A
4	REFNO2	I.32	$0 \leq$ REFNO2 $\leq 2^{32} – 1$	N/A
8	REFNO3	I.32	$0 \leq$ REFNO3 $\leq 2^{32} – 1$	N/A

2.0 Color Space

The JPEG processes in ISO/IEC 10918-1 are color blind. In this Appendix, two types of imagery are specified, 2 to 16 bit gray scale and 24 bit RGB color. For NSIF, the IREP and IREPBAND fields (defined in STANAG 4545) within the NSIF image subheader are used to identify the color space for each component present in the image; these components may be interleaved or not. When the components are interleaved, the interleave order shall be R, G, B with each MCU containing three data units, one from each component. In the non-interleaved case, each MCU consists of just one data unit from any of the components.

3.0 APPn Marker Usage

3.1 NSIF APP_6 Application Data Segment

3.1.1 NSIF requires the use of an NSIF APP_6 application data segment. This APP_6 application data segment may be identified by the null-terminated (0x00) string "NSIF" immediately following the length parameter Lp (table 39). The NSIF application data segment shall immediately follow the first SOI marker in the Image Data Field. The NSIF 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 NSIF image sub-header (COMRAT, IREPBAND, NBPP, etc.). For a description of the fields in the APP_6 marker segment see STANAG 4545.

3.1.2 Since no default Huffman tables are defined in this standard, the tables to be used by the decoder must always be present in the compressed stream. The Huffman table specification can optionally be embedded in the NSIF application data segment (shaded area in table 39). Multiple Huffman tables may be specified (up to three) in the application data segment. In this case, the table(s) will provide "default" table specification(s) for subsequent image blocks (for an explanation of image blocks see STANAG 4545). The DHT marker segment need not be embedded in the APP_6 data segment and may appear in the appropriate places in the bitstream as specified in ISO 10918-1.

3.1.3 Only DHT marker segments embedded in APP_6 will be considered defaults. Huffman tables defined outside of APP_6 are considered "custom" tables. NSIF does not allow the carryover of custom Huffman tables from one image block to the next. Custom tables must be included in each block where default tables are not used. Any Huffman table defined with a previously used table identifier shall replace the previously defined table. The format is shown in Table A-4-39 with the Huffman table segment variable fields specified in Table A-4-40 for the different image types. If no DHT marker segment is embedded in the APP_6 data segment, the length parameter, Lp, shall be equal to 20.

3.1.4 A second variation of the APP_6 application data segment is given in Table 41. Here the length of the APP_6 data segment equals that of the NSIF lossy JPEG APP_6 data segment. The length parameter, Lp, is always equal to 25. Zero (NULL) byte padding is used to achieve this length. This variation of the NSIF profile is identical to that described above with the exception that Huffman tables (DHT marker segment) may not appear in the APP_6 data segment.

Table A-4-39 - NSIF APP_6 Application Data Segment

Offset	*Field Value*	*Field Name*	*Length (bytes)*	*Comments*
0	0xFFE6	APP_6	2	NSIF application data marker.
2	see Table C-4-40	Lp	2	Segment length (2+length of application data)
4	0x4E49 0x5446 0x00	Identifier	5	Null terminated string: "NSIF"

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 of this field are defined to be consistent with ISO IS 10918-2. 14 – Sequential lossless	
20	0xFFC4	DHT	2	Define Huffman table marker	
22	see Table C-4-40	L_h	2	Length of parameters	
24	See Table C-4-40	T_c T_h	1	T_c: Table class = 0 T_h: Huffman table identifier (0-2).	first table
25	0-255	L_i	16	Number of codes of each length (BITS array)	first table
41	0-255	$V_{i,j}$	see Table C-4-40	Symbols (HUFFVAL array)	first table
		T_c T_h	1	T_c: Table class = 0 T_h: Huffman table identifier (0-2).	last table
	0-255	L_i	16	Number of codes of each length (BITS array)	last table
	0-255	$V_{i,j}$	see Table C-4-40	Symbols (HUFFVAL array)	last table
	0	Flags	2	Reserved for future use.	

Table A-4-40 - APP_6 and DHT Lengths

Field Name	*N-bit gray scale* $N \in [2, 3, \ldots, 15]$	*16-bit gray scale*	*RGB color (N = 8)*	
L_p	20, 22 + L_h	20, 58	20, 22 + L_h	
L_h	19 **+** m_t	36	$2 + \sum_{t=1}^{n} (17 + m_t)$	
T_cT_h	0x00	0x00	0x0X, $\mathbf{X} \in [0, 1, 2]$	
# of $V_{i,j}$ (m_t)	m_t **=** N + 1	17	m_t **=** 9	Predictors 1-3 and 7
	m_t **=** N + 2	17	m_t =10	Predictors 4-6

Table A-4-41 - NSIF APP_6 Application Data Segment (Second Type)				
Offset	*Field Value*	*Field Name*	*length (bytes)*	*Comments*
0	0xFFE6	APP_6	2	NSIF application data marker.
2	25	L_p	2	Segment length (2+length of application data)
4	0x4E49 0x5446 0x00	Identifier	5	Null terminated string: "NSIF"
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 of this field are defined to be consistent with ISO IS 10918-2. 14 - Sequential lossless
20	0x00	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.
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, [½, ½] filter

Table A-4-41 - NSIF APP_6 Application Data Segment (Second Type)				
Offset	*Field Value*	*Field Name*	*length (bytes)*	*Comments*
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, [½, ½] filter
25	0	Flags	2	Reserved for future use.

4.0 NSIF0003.A APP_7 Directory Data Segment

4.0.1 NSIF applications may use an NSIF0003.A APP_7 directory segment. This APP_7 application data segment may be identified by the null-terminated (0x00) string "NSIF0003.A" immediately following the length parameter Lp (table 42). The directory segments are used to provide random access to the variable length compressed data segments. These segments contain a directory of offset information for a series of scans or restart intervals depending on the directory type. In all cases, offsets are measured from the beginning of the Image Data Field in the NSIF file to the beginning of the element. The number of entries depends on the directory type and is the number of (restart intervals per scan) or (scans per block) for directory types: 'R' and 'S', respectively. The format is shown in Table A-4-42. The number of directory entries can be very large for restart interval directories.

4.0.2 In these cases, it is possible for a directory to exceed the, 64 kbyte, segment limitation imposed by the 2 byte Lp field offset in any JPEG application data segment. Since each element requires 4 bytes in the directory, this translates to a maximum of 16,379 entries. When a logical directory contains more than 16,379 elements, they must be split between more than one directory. In this case, multiple directory segments must follow each other with no other intervening data and they must be of the same directory type (restart interval). Each additional directory contains those elements, in the same order, that would have been present in the directory had there been no size limitation.

4.0.3 Another mechanism called, blocked image masking, may be used in the NSIF data format to provide direct access to image blocks, in the same spirit that directory segments provide access to entropy coded data. Blocked image masking requires the use of an image data mask subheader in the NSIF file. The content, structure and use of block image masking may be found in STANAG 4545.

Table A-4-42 - NSIF APP_7 Directory Segments				
Offset	*Field Value*	*Field Name*	*Length (bytes)*	*Comments*
0	0xFFE7	APP_7	2	NSIF directory segment marker.
2	4N + 16	L_p	2	Segment length (2 + length of application data).
4	0x4E495446 0x30303033 0x2E4100	Identifier	11	Null-terminated string “NSIF0003.A”.
15	0x52, 0x53	Directory Type	1	Directory type. Two values are defined at this time. ‘ R’ - Restart Interval Directory ‘ S’ - Scan Directory
16	1-16379	N	2	Number of directory entries. Note 0 is not allowed. Maximum value of N

				(16,379) maximizes L_p at 65532.
18		1st Offset	4	Offset to first element in this directory (restart interval, scan).
22		2nd Offset	4	Offset to second element in this directory.
4N + 14		Last Offset	4	Offset to last element in this directory.

5.0 Control Procedures

The control procedures for encoding and decoding an image using this Appendix may be found in ISO/IEC 10918-1. It is required that within STANAG 4545-compliant files, an NSIF APP_6 application data segment be placed in the compressed data stream. This data segment immediately follows the first SOI marker in the Image Data Field (Figure A-4-1). This profile also requires the use of restart intervals for the purposes of error confinement and data resynchronization. NSIF compressed imagery may include an optional APP_7 directory segment in the JPEG data

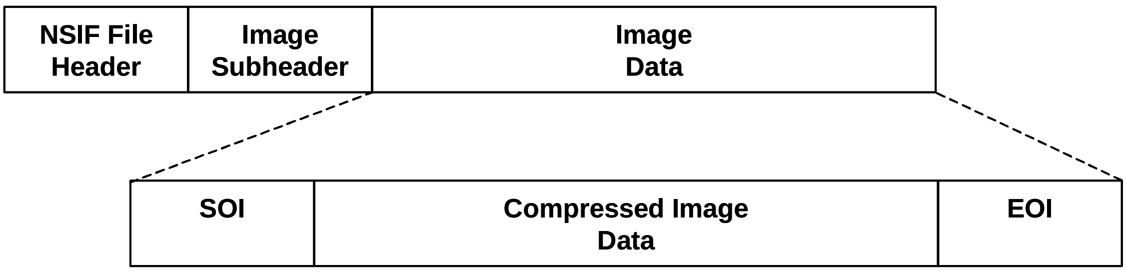

Figure A-4-1 - NSIF File Structure

6.0 File Format

6.1 Format of a JPEG Compressed Imaged Within an NSIF File

The format for NSIF image data compressed with the sequential lossless JPEG mode differs based on the number of blocks, bands, and IMODE value (B, P, S, see STANAG 4545). These different cases are described below.

6.2 Single Block JPEG Compressed Format

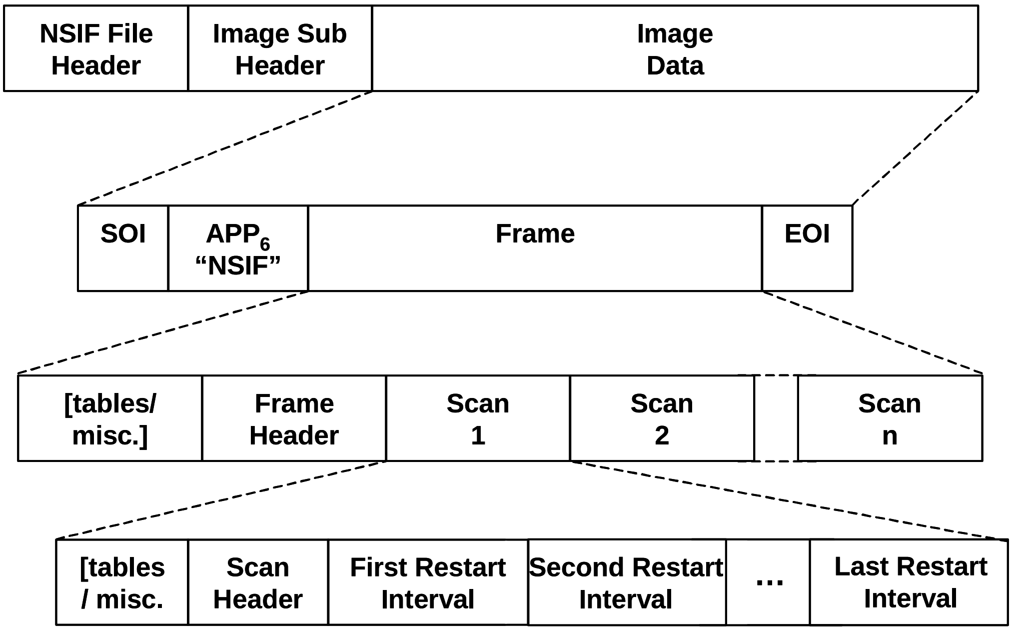

Figure A-4-2 - NSIF Single Block File Structure (IMODE=B or P)

The format for NSIF single block image data compressed with the sequential lossless JPEG mode is shown in Figure A-4-2**.**

6.3 Single Block Image Data Format

The top level of Figure A-4-2 specifies that the JPEG compressed data is contained in the Image Data Field of the NSIF file. The second level of Figure A-4-2 specifies that the single block image format shall begin with an SOI marker, shall contain one frame, and shall end with an EOI marker. Between the SOI/EOI marker pair, the data stream is compliant with ISO/IEC 10918-1 subject to the requirements and constraints of this profile.

6.4 Frame Format

The third level of Figure A-4-2 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. NSIF does not allow the use of the JPEG DNL segment which, when present, would follow the first scan in the frame.

6.5 Scan Format

The fourth level of Figure A-4-2 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 NSIF 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.

7.0 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. If the image were encoded as a single interval, then 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 as specified in ISO/IEC 10918-1. In the ISO/IEC restart intervals are optional, but NSIF requires the use of restart marker codes with a restart interval which is a multiple of the number of MCUs per row and not exceeding a maximum of 8 sample rows. Byte alignment is achieved between restart intervals per ISO/IEC 10918-1.

8.0 Multiple Block JPEG Compressed Format

The format for NSIF multiple block image data compressed with the sequential lossless JPEG mode is shown in Figure A-4-3 for IMODE=B or P. The corresponding format when IMODE=S is shown in Figure A-4-4.

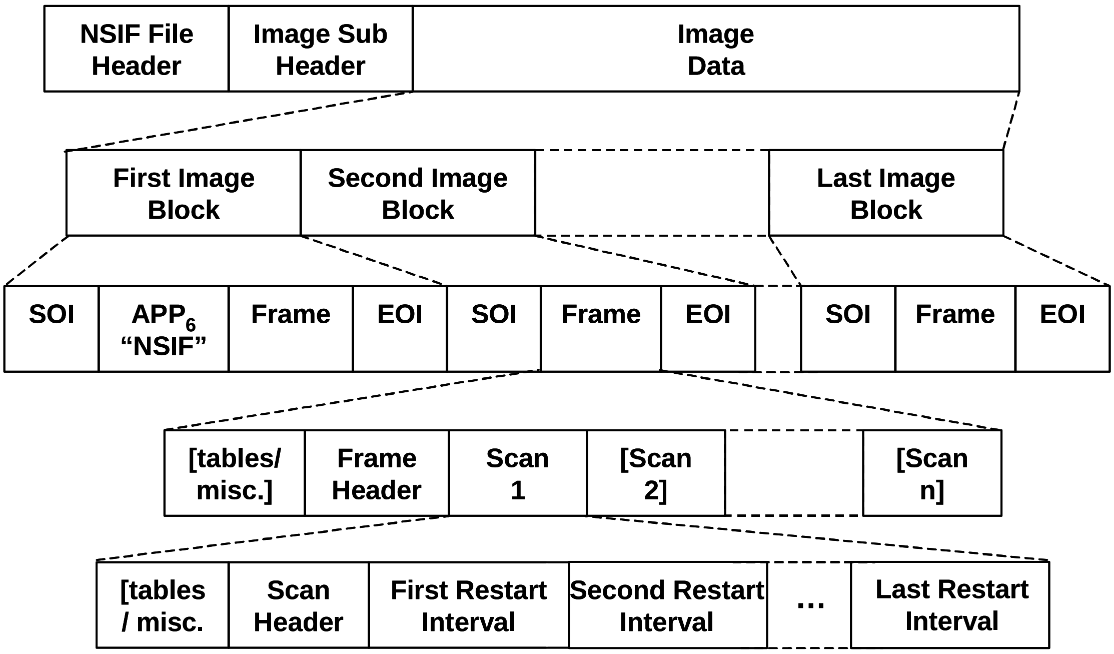

Figure A-4-3 - NSIF Multiple Block File Structure (IMODE=B or P)

8.1 Multiple Block Image Data Format (IMODE=B or P)

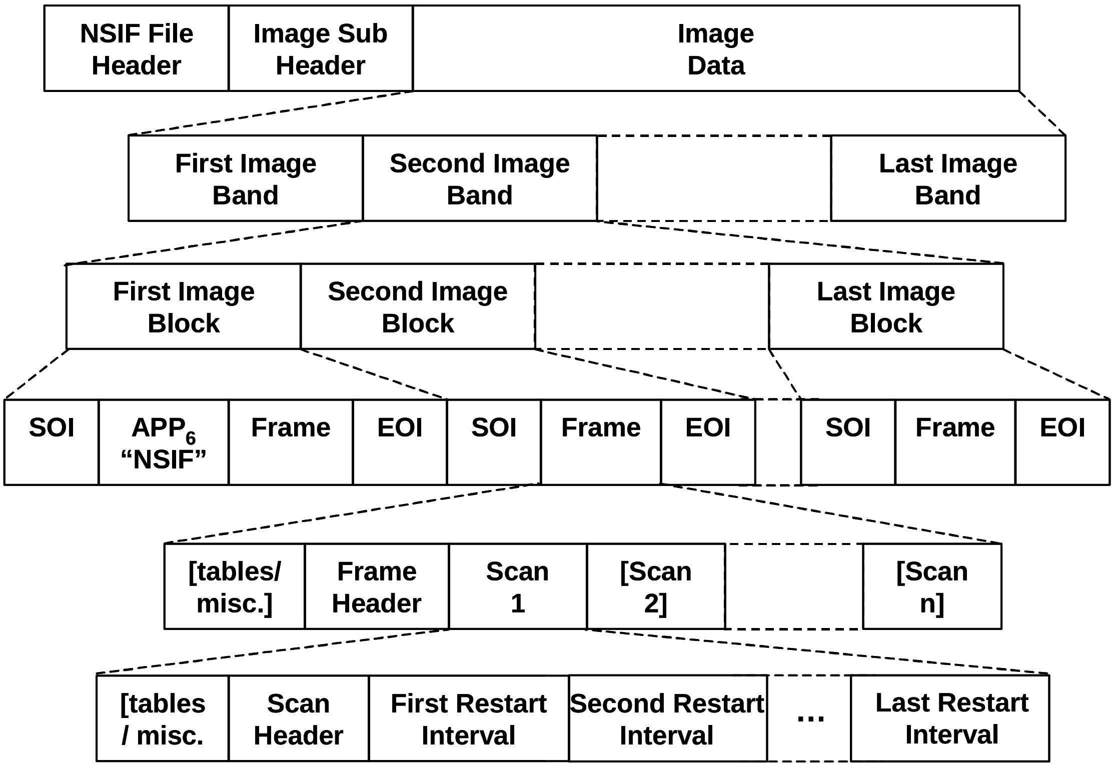

Figure A-4-4 - NSIF Multiple Block File Structure (IMODE=S)

The top level of Figure A-4-3 specifies that the JPEG compressed data is contained in the Image Data Field of the NSIF file. The second level of Figure A-4-3 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 A-4-3 specifies that each compressed block shall begin with an SOI marker, shall contain one frame, and shall end with an EOI marker.

8.2 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 A-4-4 specifies that the JPEG compressed data is contained in the Image Data Field of the NSIF file. The second level of Figure A-4-4 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 for each image band, one after the other, first to last. The third level of Figure A-4-4 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 A-4-4 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 with each frame containing only one scan that contains the compressed data from only one band.

9.0 Similarities With ISO/IEC 10918-3 "Simple Tiling"

In ISO/IEC 10918-3, extensions to the JPEG processes of ISO/IEC 10918-1 are defined. One of these extensions deals with the tiling (blocked images in NSIF terminology) of images. Of the tiling formats present in ISO/IEC 10918-3, simple tiling, is conceptually equivalent to the blocked image concept in NSIF. It is important to note that the bitstreams generated by simple tiling in ISO/IEC 10918-3 and blocked images in NSIF are not compatible. In ISO/IEC 10918-3, simple tiled images are treated as multiple frames within a single SOI/EOI marker pair. Image blocks in NSIF are treated as separate images, each within their own SOI/EOI marker pair. Within the SOI/EOI marker pairs each image block data stream conforms to ISO/IEC 10918-1 subject to the requirements and constraints of this Appendix.

Annex A, Appendix 5
Downsampled JPEG Implementation Considerations[1]

1.0 Introduction

The specification for downsampled JPEG is the standardized result of field trials. This approach provides a means to use existing lossy JPEG capabilities in the field to get increased compression for use with low bandwidth communications channels. This gives the field a very cost-effective approach for a critical capability during the period that the JPEG 2000 solution is being resolved. This technique specifically correlates to a selection option (Q3) within downsample JPEG that provides a very useable tradeoff between file compression and the resulting loss in quality.

2.0 General Requirements

2.1 Interoperability

This Appendix is intended to enable the interchange of 8-bit (Type 1) gray scale imagery compressed with the downsample JPEG algorithm. This algorithm is based on the JPEG sequential DCT image compression algorithm as described in ISO 10918-1, Digital Compression and Coding of Continuous-tone Still Images. The algorithm uses a specified downsampling filter prior to JPEG sequential DCT compression and a specified upsampling filter following JPEG decompression. This Appendix establishes the requirements for the communication or storage for interchange of image data in compressed form. Each type of operation defined by this Appendix consists of three parts:

- The compressed data interchange format which defines the image data field of the NSIF file format
- The encoder
- The decoder

This profile defines two types of operation:

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

[1] This guidance is taken from U.S. NIMA Document N-0106-97.

2.2 Encoders

Encoders shall output a full interchange format that includes the compressed image data and all table specifications used in the encoding process as illustrated in Figure A-5-1.

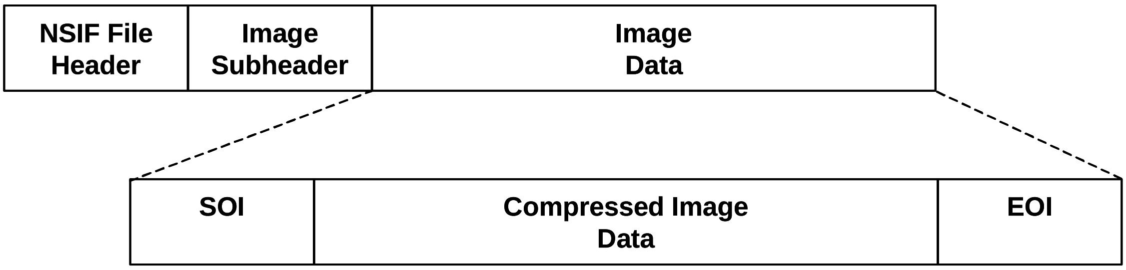

Figure A-5-1 - NSIF File Structure

2.2.1 Image Downsampling

The downsample JPEG algorithm encoder utilizes a downsampling procedure to extend the low bit-rate performance of the NSIF JPEG algorithm described in Appendix 2 of this document. Figure A-5-2 illustrates the concept. The downsampling preprocessor allows the JPEG encoder to operate at a higher bit-rate on a smaller version of the original image while maintaining an overall bit-rate that is low.

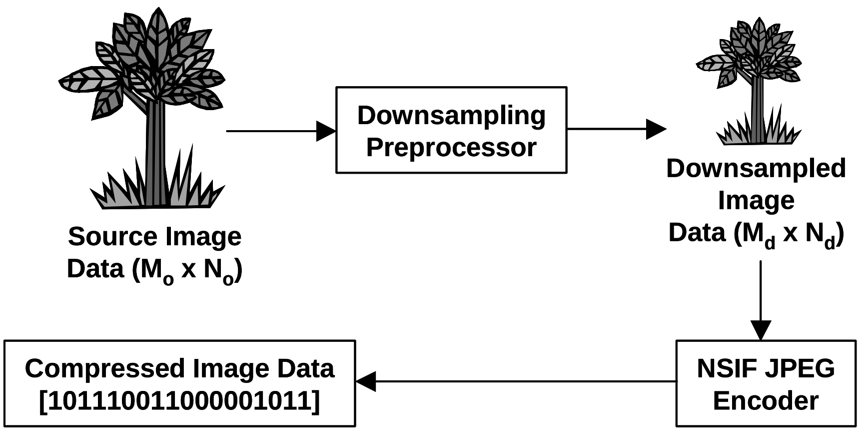

Figure A-5-2 - Downsampled JPEG Algorithm Encoder

2.2.2 JPEG Encoding

Once downsampling of the original image is performed, the encoding process is identical to that of NSIF JPEG lossy compression algorithm. Minor variations exist in the compressed data format as described in this document. This JPEG algorithm is a profile of the lossy DCT-based encoding algorithm found in ISO 10918-1. Encoders conforming to this Appendix may use any procedure in ISO 10918-1 applicable to DCT encoding subject to the requirements and restrictions expressed in Appendix 2 and herein.

2.3 Decoders

All decoders shall interpret both the full interchange format and the abbreviated interchange format.

2.3.1 JPEG Decoding

The downsample JPEG algorithm decoder decodes the compressed image data using the NSIF JPEG decoder (see Figure A-5-3). This results in reconstructed image data whose dimensions match that of the downsampled image data in Figure A-5-2.

2.3.2 Image upsampling

An upsampling postprocessor is used to return the reconstructed image data to the same dimensions as that of the original. It is important to note that the down/ upsampling processes are not lossless. The downsample JPEG algorithm makes a tradeoff between JPEG and down/upsampling artifacts in the reconstructed imagery.

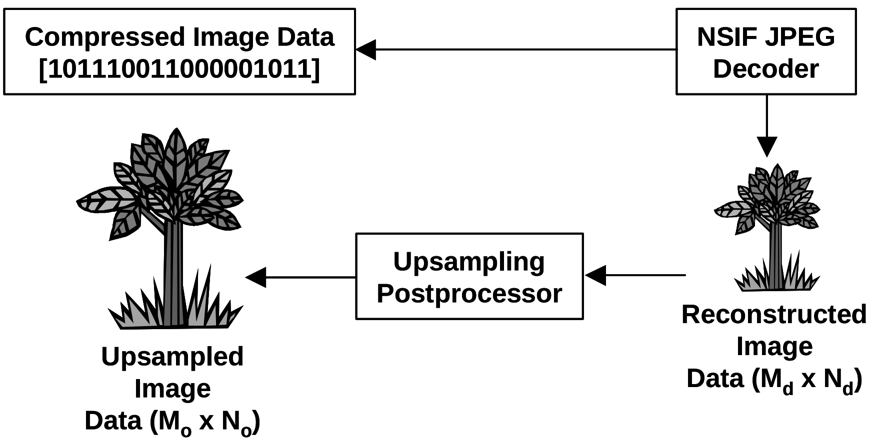

Figure A-5-3 - Downsampled JPEG Algorithm Decoder

2.4 Interchange Format-Encoders

2.4. Encoders shall output either a full or abbreviated interchange format. The full interchange format includes the compressed image data and all table specifications used in the encoding process. This capability allows for smaller files, but requires continual maintenance and dissemination of prepositioned default tables which can not be reliably achieved. Currently, non-embedded default tables are permitted by this profile. However, future compliant systems will be required to embed all necessary tables in the compressed data stream. Implementors are strongly encouraged to avoid usage of default tables.

2.4.1 The tables given in this document are recommended tables. The visible imagery 8-bit tables are allowed as defaults. Applications are free to develop tables more appropriate to their imagery than those described here. Any such custom tables must be embedded in the data stream.

2.5 Further General Requirements

Further requirements regarding the JPEG lossy compression algorithm apply to this profile. In the event of a conflict between the text of this Appendix and the base standard, this Appendix shall take precedence.

3.0 Detailed Requirements

3.1 Image Downsampling Process

Downsampling is the process of reducing the size of an image relative to the original. In this Appendix, downsampling is performed by simultaneously filtering the image and selecting a subset of the total samples available from the original. The output image will have fewer pixels and reduced dimensions, but will still be recognizable as the original image. Inputs into the downsampling module are the original image and a downsample ratio that relates the total number of pixels in the original image to the desired downsampled image. The output of this module is delivered to a compliant JPEG module for further compression and formation of a coded bitstream. The calculation of the downsampled image dimensions and adjusted downsample ratios are discussed in Section 3.1.1. The mechanics of the one-dimensional filtering operation are explained in Section 3.1.2, while the necessary equations to calculate the filter parameters are given in Section 3.1.2.1. Section 3.1.3 describes in general how the filtering operation is applied to images.

3.1.1 Downsampled Image Dimensions

The downsampled image will have reduced dimensions with respect to the original image. The number of rows and columns of the downsampled image must be calculated separately in order to properly account for non-square images. Since the downsampled image data will be compressed with JPEG, greater coding efficiency can be obtained by tuning the downsampling ratio to create downsampled image dimensions which are integer multiples of 8. This prevents the JPEG algorithm from padding blocks at image edges and wasting bits. The following equations calculate the proper downsampled image dimensions for maximum JPEG coding efficiency:

Downsampled image rows:

$$N_d = 8 \times round\left(\frac{N_o}{8 \times \sqrt{R_o}}\right)$$

Downsampled image columns:

$$M_d = 8 \times round \left(\frac{M_o}{8 \times \sqrt{R_o}} \right)$$

where N_o and M_o are the number of rows and columns in the original image respectively, and R_o is the desired downsample ratio. The actual downsample ratio that is to be used in subsequent processing should be altered to reflect the fact that truncation is performed to obtain the downsampled image dimensions. For this situation, the downsample ratio must be calculated separately for the row and columns dimensions.

Downsample ratio for the row dimension:

$$R_{row} = \frac{N_o}{N_d}$$

Downsample ratio for the column dimension:

$$R_{col} = \frac{M_o}{M_d}$$

3.1.2 Downsampling Filter Operation

The downsampled image is formed by performing separable one-dimensional filtering on the rows and columns of the original image. The filtering operation is described in the following equation as the weighted average of samples.

Filtering equation for downsampling:

$$y_i = \sum_{j=b_i}^{e_i} w_{ij} \times x_j$$

where y_i denotes a sample in the output image, x_j denotes a sample in the input image, b_i and e_i specify integer limits on the summations, and w_{ij} is the filter coefficient associated with output sample, i, and input sample, j. Both variables i and j initial values are zero. When filtering is performed in the row dimension, then y_i and x_j refer to row samples; when filtering is performed in the column dimension, then y_i and x_j refer to column samples. The equation is applied similarly for all elements in a single dimension using the same set of parameters, e_i , b_i , and w_{ij} , so no designation has been made for the particular row or column that is filtered. However, the integer limits and the filter coefficients must be calculated separately for the rows and columns when the image is non-square. NOTE: It is important not to round or truncate to an integer value (i.e., use floating-point math) when calculating W_{ij} and intermediate values of Y_i. The value of Y_i must be rounded to the nearest integer after both the horizontal and vertical downsampling is complete. The value of Y_i is restricted to be no less than 0x00 and no greater than 0xFF. Any values greater than 0xFF must be set to 0xFF and any value less than 0x00 must be set to 0x00.

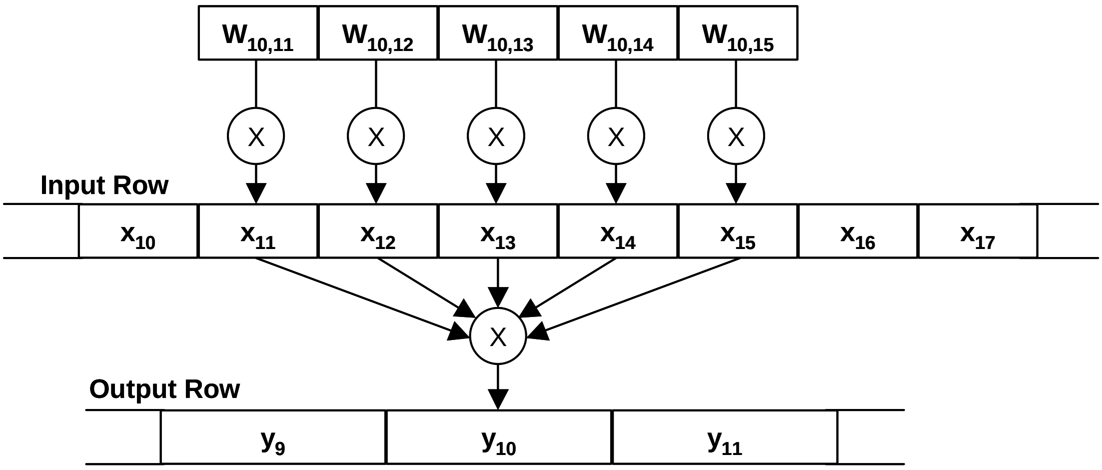

Figure A-5-4 - Downsampling Demonstration

The filtering operation is illustrated in Figure A-5-4 for the row processing case with i = 10, b_{10} = 11, e_{10} = 15, and a downsample ratio, R = 1.3.

3.1.2.1 Downsample Filter Parameter Calculations

The pertinent parameters that are required for implementation of the downsampling filter are the integer summation limits, b_i and e_i , and the coefficients, w_{ij} . The calculation of these parameters differs slightly depending on the dimension that is considered, due to the change in downsample ratio as discussed in Section 3.1.1. However, one set of parameters can be used for all the elements in the associated dimension (e.g. one set of row parameters can be applied to all the rows).

The summation limits can be calculated as shown in the following equations.

Filter beginning index: $b_i = ceil\ (\beta_i - (\alpha \times R) / 2)$

Filter ending index: $e_i = floor\ (\beta_i + (\alpha \times R) / 2)$

where:

$\alpha = 4$

$R = \begin{cases} R_{row} & \text{for row processing} \\ R_{col} & \text{for column processing} \end{cases}$

$\beta_i = (i \times R) + (0.5 \times R) - 0.5$

The parameter "α", is a value that specifies a fixed filter length, while R refers to the downsample ratios discussed in Section 3.1.1. B_i is a variable describing the location of the filter center relative to the input samples.

The filter coefficients, w_{ij} can be calculated in a two-step process.

Filter coefficients:

$$w_{ij} = \frac{c_{ij}}{\sum_{j=b_i}^{e_i} c_{ij}}$$

where:

$$c_{ij} = \cos\left[\frac{\pi(\beta_i - j)}{\alpha \times R}\right] \times \text{sinc}\left[\frac{\pi(\beta_i - j)}{R}\right]$$

and:

$$\operatorname{sinc}(x) = \begin{cases} \dfrac{\sin(x)}{x} & \text{for } x \neq 0 \\ 1 & \text{for } x = 0 \end{cases}$$

NOTE: The cosine term should never be negative, therefore the square-root term should always be valid.

3.1.3 Application of the Downsampling Filter

One-dimensional filtering is applied repeatedly along each dimension until all samples in the downsampled image have been computed. Filtering along each dimension is performed independently. One dimension is processed entirely before continuing to the complementary dimension. After processing one dimension, an intermediate image is formed as the input for processing in the other dimension. Note that the processing order (e.g. rows then columns, or vice versa) can be chosen so as to maximize performance for a given system platform. These concepts are further described in Figure A-5-5 which shows the control procedure for the downsampling operation for the example of row-column order processing.

3.1.3.1 Downsampling Along the Image Edges

In the course of downsampling an image, input values are needed that lie outside the original image. This occurs at the top, bottom, left, and right edges of the image. When extra data is needed, enough samples shall be generated by mirroring values from within the image so that the filter coefficients will always coincide with actual image samples. The mirroring point coincides with the input data sample that is exactly on the edge (e.g. first sample in a row when padding on the left of the image). Therefore, the edge sample is never repeated. This is illustrated in Figure A-5-6.

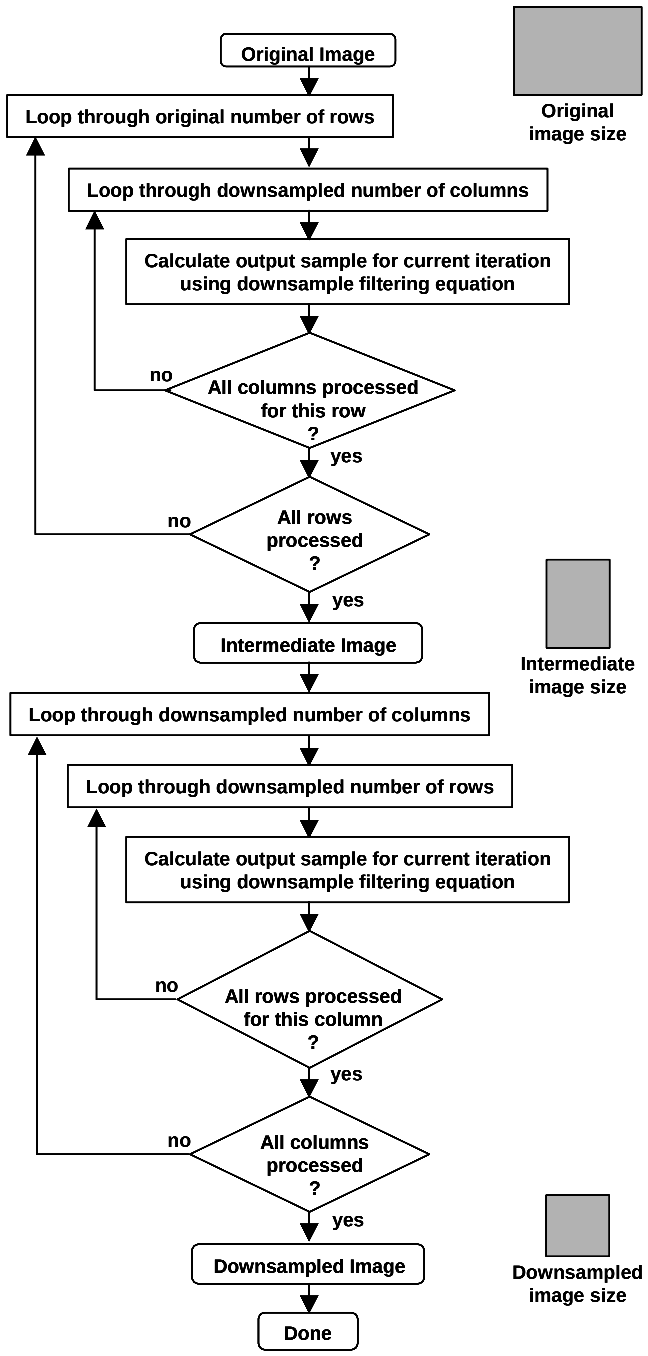

Figure A-5-5 - Control Procedure For Image Downsampling (Row-Column Order)

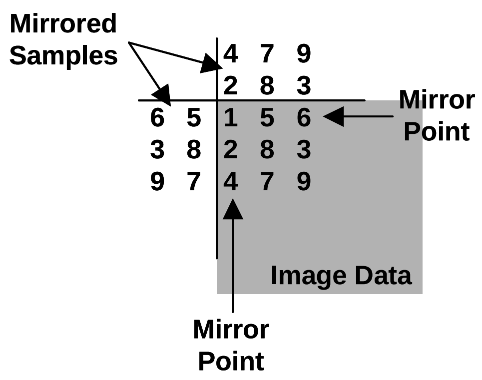

Figure A-5-6 - Illustration Of Mirroring For Image Edges

3.2 JPEG Compression of the Downsampled Image

The requirements and control procedures pertaining to the sequential DCT-based JPEG mode in Appendix B apply to this Appendix except as noted below. Once downsampling of the original image data is completed, the resultant downsampled image data is compressed with the NSIF JPEG sequential DCT lossy mode image compression algorithm. Changes in the compressed image data format are described in the following sections. The appropriate flags and parameter values for relevant fields in the NSIF image
subheader are given for the downsample JPEG algorithm in Section 3.2.2.2. Suggested Quantization and Huffman Tables for both 8-bit gray scale imagery may be found in Appendix B of ISO 10918-1.

3.2.1 Control Procedures for the Sequential DCT Lossy Mode

The control procedures for encoding an image using the JPEG sequential mode may be found in ISO 10918-1. It is required by this Appendix that an NSIF APP_6 “NSIF” application data segment be placed in the compressed data stream. This data segment immediately follows the first SOI marker in the Image Data Field (see Figure A-5-7). The format and content of this data segment are discussed in Section 3.2.2.3. Additional requirements and control procedures for NSIF JPEG sequential DCT mode may be found in Appendix B of ISO 10918-1.

3.2.2 Compressed Data Interchange Format

The interchange format consists of an ordered collection of markers, parameters, and entropy-coded data segments. A detailed description of the format is given in Appendix B of ISO 10918-1 and its references. The following sections provide the required changes that are necessary when using the downsample JPEG algorithm.

3.2.2.1 Format of a JPEG Compressed Image Within an NSIF File

The format for NSIF image data compressed with the JPEG sequential DCT lossy mode differs based on the number of blocks, bands, and IMODE value B (see STANAG 4545). These different cases are described below. Note that IMODE = S, and P are not appropriate for the down sampled JPEG algorithm since this Appendix is single band (8-bit gray scale) in nature.

3.2.2.1.1 Single Block JPEG Compressed Format

The downsampled JPEG algorithm shall be limited to original image data no larger than 2048 by 2048 pixels, single block. The format for NSIF single block image data compressed with the sequential lossy JPEG mode is shown in Figure A-5-7.

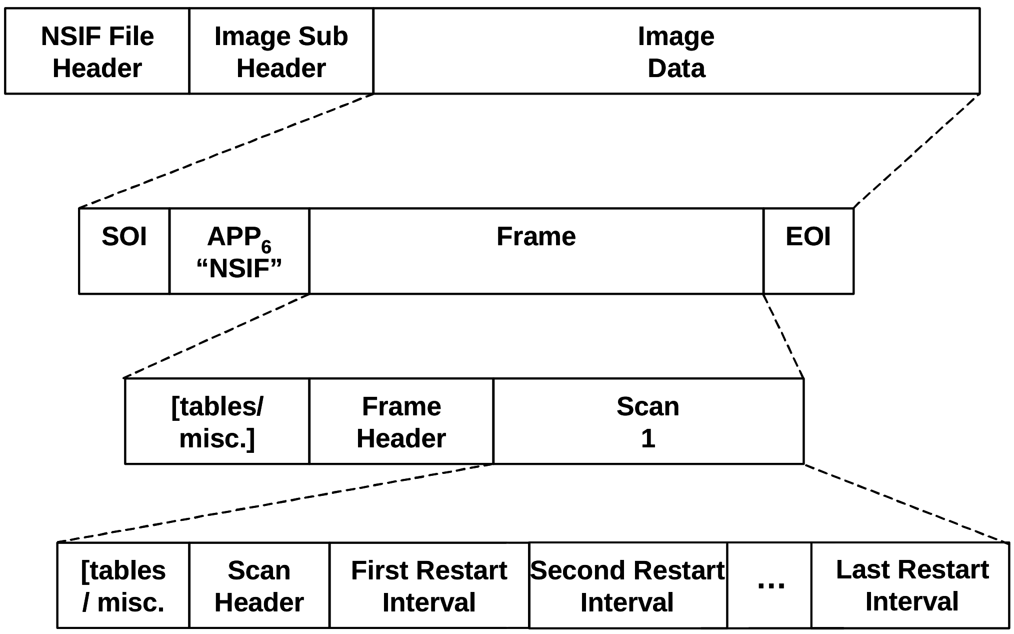

Figure A-5-7 - NSIF Single Block File Structure (IMODE=B)

3.2.2.1.1.1 Single Block Image Data Format

The top level of Figure A-5-7 specifies that the JPEG compressed data is contained in the Image Data Field of the NSIF file. The second level of Figure A-5-7 specifies that the single block image format shall begin with an SOI marker, shall contain one frame, and shall end with an EOI marker. Between the SOI/EOI marker pair, the data stream is compliant with ISO 10918-1 subject to the requirements and constraints of this profile.

3.2.2.1.1.2 Frame and Scan Formats

The frame and scan marker formats in Figure A-5-7 are the same as those found in Appendix B of ISO 10918-1 and its references. The Start-of-Frame (SOF) marker segment contains two fields "Y" and "X"which contain the number of lines and the number of samples per line in the compressed image. For the downsample JPEG algorithm, these fields shall contain the number of lines and the number of samples per line for the downsampled image data. These fields must reflect the size of the image that underwent JPEG compression.

3.2.2.1.2 Multiple Block JPEG Compressed Format

Downsampling JPEG shall not be used in conjunction with multiple blocked images. Such images must be converted to a single block, less than 2048 by 2048 pixels, then downsampled.

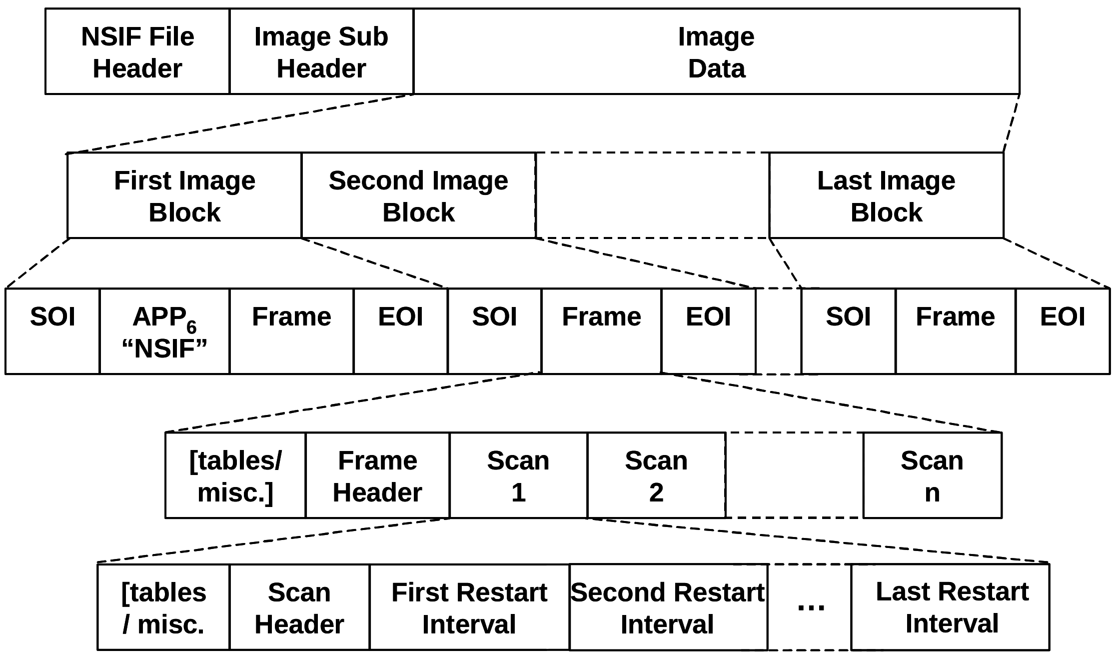

Figure A-5-8 - NSIF Multiple Block File Structure (IMODE=B or P)

3.2.2.2 NSIF Image Subheader

Fields in the NSIF image subheader must reflect the original image size. The downsample JPEG algorithm is unique in that the image and block sizes in the NSIF image subheader do not match the image or block sizes in the JPEG SOF marker data segment(s). This is necessary since the JPEG compression operates on the downsampled image or blocks while ancillary NSIF data such as overlays apply only to the original image or block dimensions. The IC field of the NSIF image subheader is set to I1. This signals a decoder that a downsampled JPEG compressed image follows.

The NROWS and NCOLS fields of the image subheader shall contain the number of significant rows and columns, respectively, in the original image. The NPPBH and NPPBV fields shall contain the number of pixels per block horizontal and the number of pixels per block vertical, respectively, of the original blocks in a blocked image. Since downsampled JPEG will only be applied to single block images NPPBH will be equal to NCOLS and NPPBV will be equal to NROWS.

The IMAG field of the NSIF image subheader is not modified for the downsample JPEG algorithm. Any decoder capable of decoding a downsampled JPEG compressed data file must restore the image and blocks to their original dimensions. The COMRAT field of the NSIF image subheader shall be set to 00.0 or 04.0. The 00.0 indicates that general purpose Huffman and Quantization Tables have been embedded into the JPEG stream. The 04.0 indicates that the specially developed tactical imagery Huffman and Quantization Tables have been embedded into the JPEG stream. These tactical tables can be found in Appendix B of ISO 10918-1.

3.2.2.3 APP_6 "NSIF" Application Data Segment

NSIF requires the use of an APP_6 "NSIF" application data segment. This application data segment shall immediately follow the first SOI marker in the image data field. The "NSIF" application data segment contains information which is needed by an interpreter but not supported by the ISO/CCITT JPEG format.

Table A-5-1 APP 6 "NSIF" Application Data Segment

Offset	*Field Value*	*Field Name*	*Length (bytes)*	*Comments*
0	0xFFE6	APP_6	2	NSIF application data marker
2	25	L_p	2	Segment length (2+length of application data)
4	0x4E49 0x5446 0x00	Identifier	5	Null terminated string: "NSIF"
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	IMODE	1	Image Format. 'B' - IMODE=B
12	0x0001	H	2	Number of image blocks per row.
14	0x0001	V	2	Number of image blocks per column.
16	0-00	Image Color	1	Original image color representation. One value is defined at this time. 0 - monochrome
17	0x08 and 0x0C	Image Bits	1	Original image sample precision.
18	0-99 0x00 0x04	Image Class	Hex	Image data class (0-99). One value is defined at this time. 00 - general purpose 04 - tactical (downsampled) imagery
19	0x01 or 0x04	JPEG Process	1	JPEG coding process. The values for this field are defined to be consistent with ISO IS 10918-2. 1 - baseline sequential DCT, Huffman coding, 8-bit sample precision
20	0x00	Quality	1	Image default Quantization & Huffman tables used. The value 0 indicates no defaults and all quantization tables must then be present in the stream.

Table A-5-1 APP 6 “NSIF” Application Data Segment

Offset	*Field Value*	*Field Name*	*Length (bytes)*	*Comments*
21	0	Stream Colour	1	Compressed colour representation. One value is defined at this time. 0 – monochrome
22	0x08 or 0x0C	Stream Bits	1	Compressed image sample precision.
23	0	Flags	4	Reserved for future use.

3.3 JPEG Decompression of the Downsampled Image

Prior to upsampling, JPEG decompression takes place resulting in a lossy reconstruction of the downsampled image data. The control procedures for decoding an image compressed with the JPEG sequential DCT lossy mode may be found in ISO 10918-1. These procedures are to be followed pursuant to the requirements of Appendix B of ISO 10918-1.

3.4 Image Upsampling Process

Upsampling is the process of increasing the number of samples through interpolation of the existing values. The process is very similar to downsampling as described in Section 3.1, but in this case the image will be sampled more frequently to increase the number of image samples. Filtering is applied to the downsampled, reconstructed image that is received from the NSIF-compliant JPEG reconstruction module. The filtering operation generates enough new samples so that the upsampled image dimensions will match the dimensions of the original image.

The upsampling process also removes the inherent offset on the downsampled image that is introduced by the downsampling process. The offset in the downsampled image is described by the following equation.

Offset in the downsampled image:

$$O_d = 0.5 - \frac{0.5}{R}$$

where *R* is the upsample ration discussed in Section 3.4.2.1. Removal of the offset is performed by subtracting out a term equal to O_d from the calculation of the upsample filter center, as described in Section 3.4.2.1. The consequence of non-compliance is a misalignment of the pixels in the reconstructed image relative to the original. Therefore, the upsampling process described in this section must always be used in conjunction with the downsampling process described in Section 3.1. This upsampling process is not recommended for use in generic interpolation applications due to the implicit assumption that the input image is offset by an amount calculated by the aforementioned equation. Calculation of the upsample ratios is discussed in Section 3.4.1. The mechanics of the one-dimensional filtering operation are explained in Section 3.4.2, while the necessary equations to calculate the filter parameters are given in Section 3.4.2.1. Section 3.4.3 describes in general how the filtering operation is to be applied to images.

3.4.1 Upsample Ratio Calculation

Separate upsample ratios must be calculated for each dimension. The two upsample ratios define the amount of expansion that is required in order to match the resolution of the downsampled image to the original image. The ratios are only dependent on the number of rows and columns in the original image, specified by N_o and M_o , and the downsampled image, specified by N_d and M_d .

Upsample ratio for the row dimension:

$$R_{row} = N_o / N_d$$

Upsample ratio for the column dimension:

$$R_{col} = M_o / M_d$$

Note that the ratios are equivalent to the downsample ratios shown in Section 3.1.1.

3.4.2 Upsampling Filter Operation

The upsampled image is formed by performing separable one-dimensional filtering on the rows and columns of the downsampled image. The mechanics of the upsample filtering process is exactly the same as the downsample case with the exception of parameter calculation. The following equation is equivalent to the filtering equation for downsampling found in Section 3.1.2, but repeated here for convenience.
Filtering equation for upsampling:

$$y_i = \sum_{j=b_i}^{e_i} w_{ij} \times x_j$$

where y_i denotes a sample in the output image, x_j denotes a sample in the input image, b_i and e_i specify integer limits on the summations, and w_{ij} is the filter coefficient associated with output sample, i, and input sample, j . When filtering is performed in the row dimension,
then y_i and x_j refer to row samples; when filtering is performed in the column dimension, then y_i and x_j refer to column samples. The equation is applied similarly for all elements in a single dimension using the same set of parameters, e_i , b_i , and w_{ij} , so no designation has been made for the particular row or column that is filtered. However, the integer limits and the filter coefficients must be calculated separately for the rows and columns when the image is non-square. The filtering operation illustrated in Figure A-5-9 for the row processing case with i = 10, b_{10} = 6, e_{10} = 9, and an upsample ratio, R = 1.3.

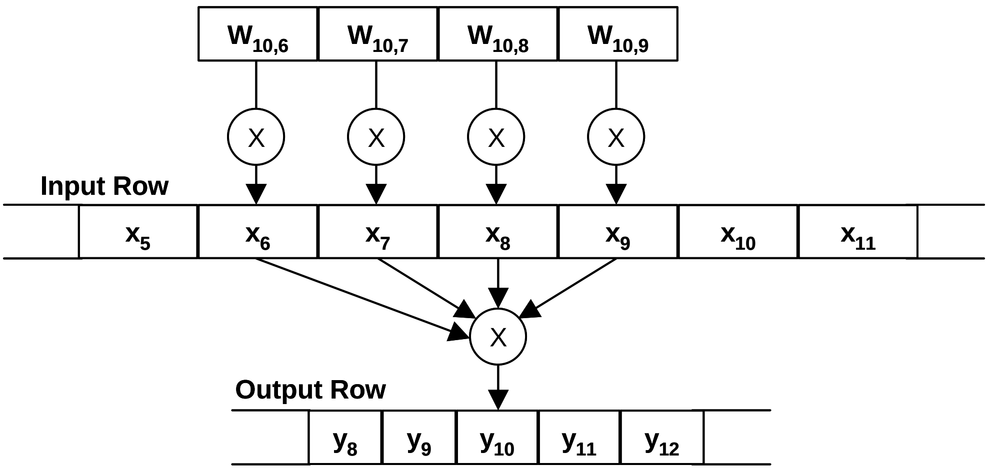

Figure A-5-9 - Upsampling Demonstration

3.4.2.1 Upsample Filter Parameter Calculations

Similar to downsampling, the parameters that require calculation are the integer summation limits and filter coefficients. One set of parameters can be applied for all the elements in the associated dimension (e.g. one set of row parameters can be applied to all the rows).

Filter beginning index: $b_i = ceil\,(\beta_i - \alpha / 2)$

Filter ending index: $e_i = floor\,(\beta i + \alpha / 2)$

where:

$\alpha = 4$

$R =$ R_{row} for row processing
R_{col} for column processing

$\beta_i = (i / R) + (0.5 / R) - 0.5$

The parameter, α, is a value that specifies a fixed filter length, while R refers to the upsample ratios discussed in Section 3.4.1. β_i is a variable describing the location of the filter center relative to the input samples. (Refer to Section 3.3 for more information concerning the upsample filter length).

The filter coefficients, w_{ij} can be calculated in a two-step process.

Filter coefficients:

$$w_{ij} = \frac{c_{ij}}{\sum_{j=b_i}^{e_i} c_{ij}}$$

where:

$$c_{ij} = \left[\cos\left(\frac{\pi\,(\beta_i - j)}{\alpha}\right)\right]^2 \operatorname{sinc}\left(\pi\,(\beta_i - j)\right)$$

3.4.3 Application of the Upsampling Filter

One-dimensional filtering is applied repeatedly along each dimension until all samples in the upsampled image have been computed. Filtering along each dimension is performed independently. One dimension is processed entirely before continuing to the complementary dimension. After processing one dimension, an intermediate image is formed as the input for processing in the other dimension. Note that the processing order (e.g. rows then columns, or vice versa) can be chosen so as to maximize performance for a given system platform. These concepts are further described in Figure A-5-10 which shows the general procedure for the upsampling operation for the example of column-row order processing.

3.4.3.1 Upsampling Along the Image Edges

In the course of upsampling an image, input values are needed that lie outside the sampling grid of the downsampled image. This occurs at the top, bottom, left, and right edges of the image. When extra data is needed, enough samples shall be generated by

mirroring values from within the image so that the filter coefficients will always coincide with actual image samples. The mirroring point coincides with the input data sample that is exactly on the edge (e.g. first sample in a row when padding on the left of the image). Therefore, the edge sample is never repeated. This is illustrated in Figure A-5-5 found in Section 3.1.3.1.

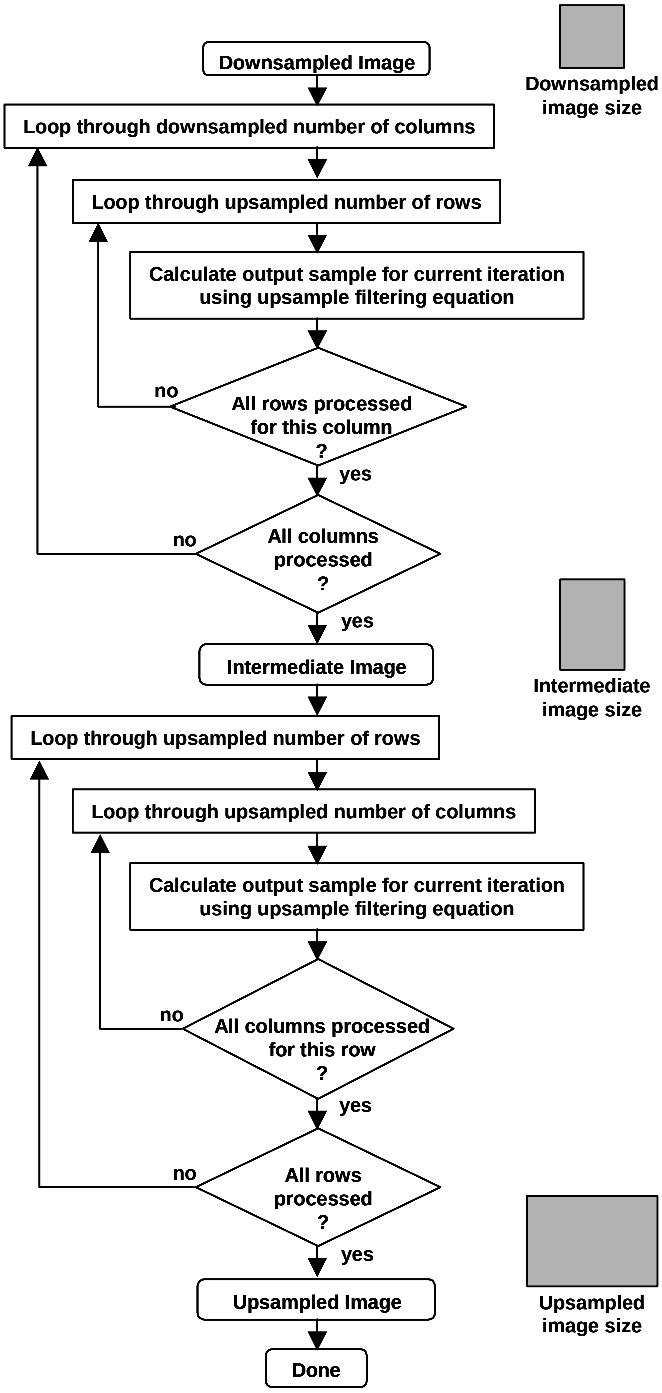

Figure A-5-10 - Control Procedure For Image Upsampling (Row-Column Order)

4.0 Notes

This section is informative only. Comments, explanations, and warnings about the compression system defined in this document are given as ancillary information. The formal requirements are outlined in Sections 2 and 3.

4.1 Reduced Complexity Image Upsampling

Certain applications may require faster upsampling at the cost of much reduced image quality. This is typically the case when computational resources are limited. Bilinear interpolation is often identified as being suitable for these applications since each interpolated value is a function of only the four nearest pixels in two-dimensions, which is a significant decrease in complexity. Furthermore, standard software routines and dedicated hardware exists to perform fast bilinear interpolation. The danger of using a standard package to perform upsampling is that the result will be a shifted version of what is expected. This is due to an offset in the downsampled image as described in Section 3.4. To ensure proper decoding, the offset must be removed during the upsampling process. The recommended method to reduce the complexity of the decoder while maintaining the proper pixel sampling positions is to reduce the filter length parameter from the value of four to two. It must be emphasized that the quality of the decoded image is noticeably worse than nominal when bilinear interpolation is used. The tradeoff between complexity and quality must be evaluated carefully before deciding to reduce the length of the upsampling filter.

4.2 Overlays

Overlays for the decoded images are intolerant to changes in image size. To prevent difficulties with overlays, the upsampled image is constrained to have dimensions equivalent to the original, uncompressed image, and the upsampling algorithm should conform to the specifications outlined in Section 3.4. If the decoder does not properly upsample the image, the overlays will be placed at incorrect locations in the image. Such an error could have serious implications for imagery users.

4.3 Inherent Quality Losses

The compression system specified in this document is aimed towards applications requiring very low bit rate compression. The nature of this requirement dictates that much information will be lost in the compression process, albeit minimization of this loss is the objective of the algorithm design. Therefore, visible distortions and resolution degradation are to be expected in the resulting images (e.g. NIIRS losses greater than 1.0 are not uncommon). It should be emphasized that this compressor is not meant for high quality compression of images. The algorithm was designed to provide images of sufficient quality to assist the decision-making process.

Annex C, Appendix 6
JPEG 2000 Implementation Considerations

1.0 PURPOSE

1.0.1 This Appendix provides guidelines for use of the new international JPEG 2000 standard when the compressing image data using this technique.

1.0.2 It should be noted that JPEG 2000 is to be used in all future development programs and is applicable to all imagery formats.

2.0 BACKGROUND

2.0.1 The international community recognized the need to provide an updated capability beyond the basic image compression capabilities of existing compression techniques. The Joint Photographic Experts Group (JPEG) had previously developed the baseline JPEG compression schemes, and went to work to develop a new technique for advanced digital imaging systems. The result was the development of JPEG 2000. This technique is applicable to all imaging systems and should be used for all future development programs to ensure interoperability of the processing and dissemination systems. Because JPEG 200 provides different resolutions and qualities from the same base compressed file, there is no longer a requirement to decompress and recompress to meet different operational requirements.

2.0.2 In all cases, the implementation of JPEG 2000 compression for either STANAG 4545 or STANAG 7023 (or others as may be developed) will comply with the profile defined for the ISR community.

3.0 DISCUSSION

The implementation of JPEG 2000 will comply with ISO/IEC 15444-1 and the BIIF profile of JPEG 2000, BPJ2K01.00. The BPJ2K Profile Implementation Guide will also be used for additional specific guidance on the use of JPEG 2000. The profile limits the options available within ISO/IEC 15444-1 to those appropriate to the ISR community.

3.1 NSIF/NITF Implementation Of JPEG 2000

Within NSIF, JPEG 2000 will comply with the requirements of both ISO/IEC 15444-1 and BPJ2K01.00. Specific guidelines for completing the fields in an NSIF file are included in AEDP-4, Annex A, Section 20.

3.2 STANAG 7023 Considerations

With STANAG 7023 files, JPEG 2000 will comply with the requirements of the ISO profile BPJ2K01.00. Specific implementation guidance for use of JPEG 2000 in STANAG 7023 files are included in the STANAG.

4.0 SUMMARY / CONCLUSIONS

This document provides general guidelines for the implementation of JPEG 2000 in NIIA systems. Users are advised to refer to ISO/IEC 15444-1, BPJ2K01.00, STANAGs 4545 and 7023, and AEDP-4 for further information. These documents are the authoritative sources and will be used for compliance testing.

ANNEX B:

File Naming Conventions

1.0 INTRODUCTION

This Annex is to provide guidance to users of the NATO Intelligence, Surveillance, and Reconnaissance (ISR) Interoperability Architecture (NIIA) on naming files. Since most systems use unique techniques for identifying files within the system architecture. In some cases, the systems simply use file name attributes to identify the type of file. In other cases, elaborate names are created that include metadata to identify the file from the name alone. This Annex identifies some of the more common techniques to be used as examples for users within the NIIA community, with the intent to not proliferate additional conventions without specific requirements.

2.0 CURRENT CONVENTIONS

There are principally two approaches to file naming conventions. One allows a freeform name with a specific extension or tag to identify the file type. The second uses the name to identify some of the metadata associated with the file, such as image location, time of acquisition, processing performed, etc. This Annex is intended to be used as a reference to identify current naming schemes, and provide guidance to users of NIIA data to minimize proliferation of additional schemes. There is no intent to mandate a specific file naming convention. Note that these naming conventions are provided for reference. If a user is provided a file with one of these names, the user will be able to decode it. Also, if a file naming convention is needed, developers are encouraged to use one of these if possible, rather than creating a new one.

2.1 File Extension Convention

2.1.1 There are four data formats for ISR data within the NIIA. While other types of data can be exchanged based on commercial standards, specific extensions should be adopted for use with files structured in compliance with the four data format standards. Data files produced by commercial applications should continue to use the extensions associated with the respective application (e.g. “*.doc” for Microsoft Word files).

2.1.2 The recommended extensions for NIIA data formatted files is as shown in the table below. Whenever possible, files stored or transported from system to system should include the appropriate extension. In the case of a file that uses one format to wrap data in another format (for example, GMTI data wrapped in the NSIF DES and file structure), the outer format – the one initially encountered in the file – should be the one used in the extension.

TABLE B-1
Recommended File Extensions For NIIA Data Formats

STANDARD	***EXTENSION***	***COMMENTS***
STANAG 4545, NATO Secondary Imagery Format (NSIF)	***.nsf	
MIL-STD-2500B, National Imagery Transfer Format (NITF)	***.ntf	Included for completeness

STANAG 4607, NATO GMTI Data Format	***.gmt	Note that frequently data is not labeled as a file in transmission due to need to minimize data rate on the transmission link
STANAG 4609, NATO Digital Motion Imagery Format	multiple	Use the commercially approved extensions for the file type (e.g. ***.mpg, ***.mxf, etc.)
STANAG 7023, NATO Primary Image Format (NPIF)	***.npf	NPIF is primarily a data transfer format, but files can be stored for later use with this extension
DIGEST	multiple	Provide DIGEST files with names in accordance with DIGEST Part 2, Annex E. (e.g. ***.txt, ***.iif, ***.dt1) [Note that this document also provides the directory structure for groups of DIGEST files.]

2.2 Name Conventions

This Annex identifies general issues and guidance related to Standard IDs and naming conventions. It also identifies community uses of each of them where known. Several alternative methods for identifying imagery products and assigning file names are in use within the NIIA community. The naming conventions addressed in this Annex are:

- Forty-Character File Naming Convention
- Sixty-four Character File Naming Convention
- Moving Target Indicator (MTI) File Naming Convention
- Airborne SDE Programs
- U.S. DCGS Conventions

2.2.1 Forty-Character File Naming Convention.

This convention is used in a number of operational systems using NSIF/NITF to provide general references to specific images without having to open the image file and parse the metadata.

2.2.1.1 Forty-Character Naming Convention General Overview.

This file naming structure consists of a forty-character image identifier followed by suffix extensions as follows:

FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF. rN_PART_nn_OF_mm.NTF

Where:
FFFF… The Forty-Character Image Identifier (see table B-2).

.NTF An optional extension to indicate the file is formatted in NITF.

.rN An extension to indicate the resolution of the image within the file.
r0 indicates the image is of original/full resolution.
Values r1 through r9 indicate reduced resolution in factors of 2 (1/2, 1/4, 1/8, 1/16, 1/32, …).

_PART_nn_OF_mm An extension to indicate that the file is instance “nn” of “mm” total files that comprise the entire product specified by the forty-character identifier. Note: Under some circumstances, the total number of files comprising the entire product may not be known when the files are initially being produced. In that case, the mm value is set to 00 to indicate the count is unknown. Where possible, the actual count should at least be placed in the last file of the product sequence.

2.2.1.2 Forty-Character Image Identifier.

The field structure of the forty-character image identifier is shown in Table B-2. The structure provides a means for uniqueness of product identification and for association of multiple files, that may comprise a single product. Each NIIA image production system using the 40-character identifier convention places additional product specific constraints on the use of the 40-character identifier. The image/product identifier Tagged Record Extension (TRE) specification applicable to the producing systems (i.e., STDIDA, STDIDB, AIMIDA, and AIMIDB) specifies these additional constraints. Maintaining the proper relationships of the identifier sub-field values with the identifier's usage in file headers, image subheaders, identification TREs, and as a file naming convention are critical to proper use and interpretation of imagery products and associated support data.

2.2.1.3 Identifier Field Value Dependency on Data Coverage.

Proper population and use of field values used in the identifier depends on the data coverage to which the specific identifier applies. The values for beginning and ending image segments and those for starting and ending block numbers shall be populated from the perspective of the entire data coverage of the identified imaging operation. When the identifier pertains to an entire imaging operation (e.g. IID2/ITITLE), the segment and block indexes shall reflect the full extent of the total data coverage. When the identifier pertains to a portion of the imaging operation (e.g. FTITLE and filename), the segment and block indexes shall reflect the relative location in the entire data coverage of the imaging operation that is included in the identified data portion.

Table B-2
40-Character Image Identifier (Generic)

Position	***Title***	***Description***	***Range***
1-7	Image/Product Date	The date representing the currency of the image product data; the date the image data was acquired. This date shall be the same as the date recorded in the NITF image subheader IDATIM field.	DDMONYY Note: Month uses three character month.
8-11	Mission Number, Primary	An alphanumeric code that identifies the collection means for the imagery product. e.g. mission project number, DIA-assigned Project Code, aircraft identifier, etc. The allowed values are constrained to the alphanumeric value range or value list specified for the applicable collection system and its associated production system.	Mission and/or collection system specific. See the specification for the applicable product identification TRE.

Table B-2
40-Character Image Identifier (Generic)

Position	***Title***	***Description***	***Range***
12-13	Mission Number, Secondary	An alphanumeric code that refines/expands the mission number by providing an 'instance' sequence. e.g. a flight number, a fly-over index/count, a re-visit number, etc.	01 – 09 A1 to A9 B1 to B9 … Z1 to Z9 00
14-16	Image Operation Number	The index value (count) of the acquisition or collection task/objective that resulted in this product.	000 to 999
17–18	Beginning image segment ID	A code used in conjunction with other fields in the identifier to characterize multi-segment products. - For single-segment products, the value is always 'AA'. - For multi-segment products, the value depends on the scope/coverage of the identifier. - When the identifier refers to the entire imaging operation, the code is 'AA'. - When the identifier refers to a portion of the imaging operation, the code is that of the segment to which the first pixel value in the portion belongs.	AA to ZZ
19–20	Reprocess number	A code to differentiate different instances of the same image product resulting from reprocessing of the source data and/or enhancement processing of the originally processed image data. The value '00' indicates the data is the originally processed image. Values in the range '01' through '99' represent subsequent instances of reprocessing or enhancement processing.	00 to 99
21–23	Replay	Replay indicates whether the data was retransmitted or re-stored to overcome exchange errors. Its value allows differentiation among multiple transmissions or exchanges of the same image product. The value '000' indicates that the data is from the initial exchange. Values in the range 'T01' to 'T99' indicate the instance of retransmission. Values in the ranges 'P01' to 'P99' and 'G01' to 'G99' are reserved for future use.	000, G01 to G99, P01 to P99, T01 to T99
24	Reserved for system specific use.	The default values for this field are Underscore '_' and the Space character. When using the identifier as a file name, the underscore character shall be used. Either may be used when the identifier is used within NSIF subheader or SDE fields.	Underscore "_" Space Character

Table B-2
40-Character Image Identifier (Generic)

Position	***Title***	***Description***	***Range***
25–26	Starting Column Block (or tile) Number	The NITF block column index number for the first block of the image segment present in the actual data coverage to which the identifier is applicable. The column count is relative to the start of the segment specified by the Beginning Image Segment ID. - The value depends on the scope/ coverage of the identifier. - When the identifier refers to the entire imaging segment, the code is ‘01’. - When the identifier refers to a portion of the imaging segment, the code is the column index of the block to which the first pixel value in the data coverage belongs. - For single block images this field shall contain 01.	01 - 99
27	Flag1	Reserved for system specific indicator flag. Default value is “0”	0
28–31	Starting row block (or tile) number	The NITF block row index number for the first block of the image segment present in the actual data coverage to which the identifier is applicable. The row count is relative to the start of the segment specified by the Beginning Image Segment ID. - The value depends on the scope/coverage of the identifier. - When the identifier refers to the entire imaging segment, the code is ‘01’. - When the identifier refers to a portion of the imaging segment, the code is the row index of the block to which the first pixel value in the product coverage belongs. - For single block images this field shall contain 00001.	0001 - 9999
32–33	Ending Image Segment ID	For single-segment products, the value is always ‘AA’. For multi-segment products, the value depends on the scope/coverage of the identifier. When the identifier refers to the entire imaging operation, the code is that of the last segment in the imaging operation. When the identifier refers to a portion of the imaging operation, the code is that of the segment to which the last pixel value in the portion belongs.	AA to ZZ

Table B-2
40-Character Image Identifier (Generic)

Position	***Title***	***Description***	***Range***
34–35	Ending column block (or tile) number	The NITF block column index number for the last block of the image segment present in the actual data coverage to which the identifier is applicable. The column count is relative to the start of the segment specified by the Ending Image Segment ID. - The value depends on the scope/ coverage of the identifier. - When the identifier refers to the entire imaging segment, the code is the last column index in the entire segment. - When the identifier refers to a portion of the imaging segment, the code is the column index of the block to which the last pixel value in the data coverage belongs. - For single block images this field shall contain 01.	01 – 99
27	Flag2	Reserved for system specific indicator flag. Default value is “0”	0
37–40	Ending row block (or tile) number	The NITF block row index number for the last block of the image segment present in the actual data coverage to which the identifier is applicable. The row count is relative to the start of the segment specified by the Ending Image Segment ID. - The value depends on the scope/ coverage of the identifier. - When the identifier refers to the entire imaging segment, the code is the last row index in the entire segment. - When the identifier refers to a portion of the imaging segment, the code is the row index of the block to which the last pixel value in the data coverage belongs. - For single block images this field shall contain 00001.	0001 - 9999

2.2.2 Sixty-four Character Product Identifier

This convention is used by other programs when addition information is to be contained in the file name.

2.2.2.1 Sixty-four-Character Naming Convention.

This file naming structure consists of a sixty-four-character image identifier followed by suffix extensions as follows:

FFFFFFFFFFFFFFFFFFF…..FFFFFFFFFFFFFFFFFF.rN_PART_nn_OF_mm.NTF

Where:

FFFF…	The Sixty-four-Character Image Identifier (see table B-3).
.NTF	An optional extension to indicate the file is formatted in NSIF.
.rN	An extension to indicate the resolution of the image within the file. r0 indicates the image is of original/full resolution. Values of r1 through r9 indicate reduced resolution in factors of 2 (1/2, 1/4, 1/8, 1/16, 1/32, …).
_PART_nn_OF_mm	An extension to indicate that the file is instance “nn” of “mm” total files that comprise the entire product specified by the forty-character identifier. Note: Under some circumstances, the total number of files comprising the entire product may not be known when the files are initially being produced. In that case, the mm value is set to 00 to indicate the count is unknown. Where possible, the actual count should at least be placed in the last file of the product sequence.

2.2.2.2 Sixty-Four-Character Image Identifier.

The field structure of the sixty-four character image identifier is shown in Table B-3. The structure provides a means for uniqueness of product identification and for association of multiple files, that may comprise a single product. Each NIIA image production system using the 64- character identifier convention places additional product specific constraints on the use of the 64-character identifier. The image/product identifier Tagged Record Extension (TRE) specification applicable to the producing systems (i.e., STDIDC) specifies these additional constraints.

2.2.2.3 Identifier Field Value Dependency on Data Coverage.

Proper population and use of field values used in the identifier depends on the data coverage to which the specific identifier applies. The values for beginning and ending image segments and those for starting and ending block numbers shall be populated from the perspective of the entire data coverage of the identified imaging operation. When the identifier pertains to an entire imaging operation (e.g. IID2/ITITLE), the segment and block indexes shall reflect the full extent of the data coverage. When the identifier pertains to a portion of the imaging operation (e.g. FTITLE and filename), the segment and block indexes shall reflect the relative location in the entire data coverage of the imaging operation that is included in the identified data portion.

Table B-3
64-Character Image Identifier

Position	***Title***	***Description***	***Range***
1-14	Image/Product Acquisition Date and Time	The date and UTC time representing the currency of the image product data; the date/time the image data was acquired. This date and time shall be the same as the date recorded in the NSIF image subheader IDATIM field.	YYYYMMDDhhmmss

Table B-3
64-Character Image Identifier

Position	***Title***	***Description***	***Range***
15-18	Mission Number, Primary	An alphanumeric code that identifies the collection means for the imagery product. e.g., mission project number, DIA-assigned Project Code, aircraft identifier, etc. The allowed values are constrained to the alphanumeric value range or value list specified for the applicable collection system and its associated production system.	Mission and/or collection system specific. See the specification for the applicable product identification TRE.
19-33	Mission Number, Secondary	An alphanumeric code that refines/ expands the primary mission number with further mission-specific identification. e.g., a mission number from an Air Tasking Order.	Mission and/or collection system specific. See the specification for the applicable product identification TRE.
34-35	Mission Number, Tertiary	An alphanumeric code that refines/ expands the significance of the previous two mission number fields by providing an ‘instance’ sequence. e.g. a flight number, a fly-over index/count, a re-visit number, etc.	01 – 09 A1 to A9 B1 to B9 … Z1 to Z9 00
36-40	Image Operation Number	The index value (count) of the acquisition or collection task/objective that resulted in this product.	000 to 999
41-42	Beginning image segment ID	A code used in conjunction with other fields in the identifier to characterize multi-segment products. - For single-segment products, the value is always ‘AA.’ - For multi-segment products, the value depends on the scope/coverage of the identifier. - When the identifier refers to the entire imaging operation, the code is ‘AA.’ - When the identifier refers to a portion of the imaging operation, the code is that of the segment to which the first pixel value in the portion belongs.	AA to ZZ
43-44	Reprocess number	A code to differentiate different instances of the same image product resulting from reprocessing of the source data and/or enhancement processing of the originally processed image data. - The value ‘00’ indicates the data is the originally processed image. - Values in the range ‘01’ through ‘99’ represent subsequent instances of reprocessing or enhancement processing.	00 to 99

Table B-3
64-Character Image Identifier

Position	***Title***	***Description***	***Range***
45-47	Replay	Replay indicates whether the data was retransmitted or re-stored to overcome exchange errors. Its value allows differentiation among multiple transmissions or exchanges of the same image product. - The value ‘000’ indicates that the data is from the initial exchange. - Values in the range ‘T01’ to ‘T99’ indicate the instance of retransmission. - Values in the ranges ‘P01’ to ‘P99’ and ‘G01’ to ‘G99’ are reserved for future use.	000, G01 to G99, P01 to P99, T01 to T99
48	Reserved for system specific use	The default values for this field are Underscore ‘_’ and the Space character. When using the identifier as a file name, the underscore character shall be used. Either may be used when the identifier is used within NSIF subheader or SDE fields.	Underscore “_” Space Character
49-50	Starting Column Block (or tile) Number	The NITF block column index number for the first block of the image segment present in the actual data coverage to which the identifier is applicable. The column count is relative to the start of the segment specified by the Beginning Image Segment ID. - The value depends on the scope/coverage of the identifier. - When the identifier refers to the entire imaging segment, the code is ‘01.’ - When the identifier refers to a portion of the imaging segment, the code is the column index of the block to which the first pixel value in the data coverage belongs. - For single block images this field shall contain 01.	01 - 99

Table B-3
64-Character Image Identifier

Position	***Title***	***Description***	***Range***
51-55	Starting row block (or tile) number	The NITF block row index number for the first block of the image segment present in the actual data coverage to which the identifier is applicable. The row count is relative to the start of the segment specified by the Beginning Image Segment ID. - The value depends on the scope/coverage of the identifier. - When the identifier refers to the entire imaging segment, the code is ‘01’. - When the identifier refers to a portion of the imaging segment, the code is the row index of the block to which the first pixel value in the product coverage belongs. - For single block images this field shall contain 00001.	00001 – 09999
56 - 57	Ending Image Segment ID	For single-segment products, the value is always ‘AA.’ For multi-segment products, the value depends on the scope/coverage of the identifier. When the identifier refers to the entire imaging operation, the code is that of the last segment in the imaging operation. When the identifier refers to a portion of the imaging operation, the code is that of the segment to which the last pixel value in the portion belongs.	AA to ZZ
58-59	Ending column block (or tile) number	The NSIF block column index number for the last block of the image segment present in the actual data coverage to which the identifier is applicable. The column count is relative to the start of the segment specified by the Ending Image Segment ID. - The value depends on the scope/coverage of the identifier. - When the identifier refers to the entire imaging segment, the code is the last column index in the entire segment. - When the identifier refers to a portion of the imaging segment, the code is the column index of the block to which the last pixel value in the data coverage belongs. - For single block images this field shall contain 01.	01 - 99

Table B-3
64-Character Image Identifier

Position	***Title***	***Description***	***Range***
60-64	Ending row block (or tile) number	The NITF block row index number for the last block of the image segment present in the actual data coverage to which the identifier is applicable. The row count is relative to the start of the segment specified by the Ending Image Segment ID. - The value depends on the scope/coverage of the identifier. - When the identifier refers to the entire imaging segment, the code is the last row index in the entire segment. - When the identifier refers to a portion of the imaging segment, the code is the row index of the block to which the last pixel value in the data coverage belongs. - For single block images this field shall contain 00001.	00001 - 09999

2.2.3 Moving Target Indicator (MTI) File Naming Convention

The following file naming convention is used to define files that contain MTI data without an associated image segment.

Table B-4
40-Character Image ID for MTI Files without Image Segments

Position	***Title***	***Description***	***Range***
1-3	File header field	The characters “MTI” will be used.	MTI
4–17	Date and time of collection	DATIME field from MTIRPB TRE.	YYYYMMDDhhmmss
18–37	Mission Number	AC_MSN_ID field from ACFTB	Mission Name as specified in the AIP ARS to Ground ICD Table 3.3.1.2.6-1.
38 – 40	Reserved	Fill with spaces.	spaces

2.2.4 Systems Using The Airborne Support Data Extensions

The airborne support data extensions are identified in the NSIF Registry in AEDP-4. Programs using these extensions are encouraged to review this file naming convention for applicability.

2.2.4.1 Management and Tracking of the Imaging Process.

Some imagery exploitation management systems currently depend on the 40-character product identifier to track the imaging process. Once the imaging collection process is completed, the set of files (one or more) associated with the imaging operation is placed in storage for retrieval. A message is sent to the exploitation management system that the requested imaging operation is complete and product is available for exploitation. The 40-character ID is the means for the management system to identify which file(s) resulted from the task to collect imagery. The following describes the relationship of AIMID, FTITLE, ITITLE/IID2, and the 40-character ID for management of the imaging process.

2.2.4.2 Tactical Image ID

This section provides guidance to the tactical users for file naming, FTITLE, product naming, and the relationships between them. It also provides guidance for handling derivative imagery products. According to AEDP-4, the additional image identification (AIMIDx) Tagged Record Extension (TRE):

- is a required component of all imagery files (one in each subheader of every NSIF image segment);
- is “used for storage and retrieval from standard imagery libraries”; and
- provides the forty characters to populate the ITITLE/IID2 field within the image subheader.

Airborne sensors that generate NSIF files frequently populate the AIMID TRE with the allowed default values for a variety of reasons (e.g.; lack of onboard processing capability, lack of information, etc). Several groups within the NIIA community have recognized a problem with image identification (image ID) and have attempted to address the problem separately.

Table B-5
Mapping Between AIMIDB and ITITLE/IID2

ITITLE/IID2 Location (Bytes)	***AIMIDB Field***
1 - 7	ACQUISITION_DATE (formatted as DDMMMYY, where: DD is the day of the month, MMM is a three-letter abbreviation of the month, JAN, FEB, … DEC, YY is the least significant 2 digits of the year).
8 – 11	MISSION_NO
12 – 13	FLIGHT_NO
14 – 16	OP_NUM
17 – 18	CURRENT_SEGMENT
19 – 20	REPRO_NUM
21 – 23	REPLAY
24	Space
25 – 26	START_TILE_COLUMN (least significant 2 bytes)
27 – 31	START_TILE_ROW
32 – 33	END_SEGMENT
34 – 35	END_TILE_COLUMN (least significant 2 bytes)
36 - 40	END_TILE_ROW

2.2.5 U.S. Distributed Common Ground/Surface Station (DCGS)

The DCGS community uses the image ID to coordinate imagery flow within DCGS elements (e.g., between the Common Imagery Processor (CIP), Imagery Product Library (IPL), screener and exploitation workstations). The sensors, CIP, and screeners do not have a standard method for ensuring a unique tactical image ID. The DCGS community developed an interim solution with the intent of providing the most uniqueness and functionality with the least impact/disruption while accommodating the Imagery Exploitation Support System's (IESS) twenty-four character (for uniqueness) and IPL's forty-character image ID constraints. The interim solution does not create a new AIMID TRE version. The CIP will continue to populate the AIMIDB TRE with valid data, and edit any fields containing default values, received from the sensors. For the interim solution, the CIP only changes the data sources used to populate the first forty characters of the image subheader ITITLE/IID2 field. The DCGS community plans to implement their interim solution within the CIP, IPL and IESS by December 2003 (preceded by an August 2003 demonstration). Table B-6 displays how the CIP will populate the first forty characters of the ITITLE/IID2 field to create a unique image ID.

Table B-6
U.S. DCGS Short Term Unique Image ID Solution

ITITLE/IID2 Location (Bytes)	***Current Field Name(s) (from AIMIDB)***	***New Field Name***	***New Field Name Description***
1-7	ACQUISTION_DATE	ACQUISTION_DATE	No change, reference Table 8-4 in STDI-0002, Version 2.1, 16 Nov 2000 Note: This is the image collection date and not the start of mission date or aircraft takeoff date.
8-11	MISSION_NO	MISSION_NO	No change, reference Table 8-4 in STDI-0002, Version 2.1, 16 Nov 2000 (Format = PPNN, where PP is the DIA project code and NN is the flight/sortie number)
12-16	FLIGHT_NO & OP_NUM	OP_NUM	5-char (numeric) image operation number (00000-99999).
17-18	CURRENT_SEGMENT	PRODUCER_CODE	2-char DOD/DIA producer code. Uniquely defines a producer.
19-24	REPRO_NUM, REPLAY, & Space	PRODUCT_NO	6-char "producer defined" product id number which uniquely defines each product produced by a given producer. This could be a simple one-up product sequence number. See below for example CIP definition.
25 -26	START_TILE_COLUMN	CURRENT_SEGMENT	2-char (alphanumeric) current segment ID. Same as CURRENT_SEGMENT as defined in AIMIDB.
27 - 29	START_TILE_ROW (bytes 27 - 29)	REPLAY	3-char (alphanumeric) replay indicator. Same as REPLAY as defined in AIMIDB.
30 -32	START (bytes 30 -31) & END_SEGMENT (byte 32)_TILE_ROW	PRODUCER_SN	3-char (numeric) producer serial numbers (000-999 or 000-FFF). Defines a unique instance of the producer.
33-40	END_SEGMENT (byte 33), END_TILE_COLUMN & END_TILE_ROW	PRODUCTION_DATIM	8-char (hex) production date/time (GMT represented in hexadecimal as elapsed time in seconds since midnight January 1, 1970.

The "CIP Product Number" is collectively defined as a 6-character field. It consists of three subfields: processing configuration number (1 char, 0-F), product type identification (2 chars, 01-FF), and product sequence number (3 chars, 000-FFF). Example: If the processing configuration = 1, product type identification = 12, and product sequence number = 25; then the PRODUCT_NO = 10C019 (hex).

Encryption in the NIIA

[TBD]

Note: A project is underway to define standard encryption for use in wideband applications within the NIIA. The final results will be documented in this Annex once complete.

Spectrum Management Guidelines for NIIA Systems

[TBD]

Note: A project is underway to define guidelines for managing the electromagnetic spectrum needs within the NIIA. This study will address both the data link and active sensor requirements. The final results will be documented in this Annex once complete.

ANNEX E:

Ground Station Coding and Capability Matrix

[TBD]