INTERNATIONAL STANDARD

ISO/IEC 12087-5

First edition
1998-10-15

Information technology — Computer graphics and image processing — Image Processing and Interchange (IPI) — Functional specification —

Part 5:
Basic Image Interchange Format (BIIF)

Technologies de l'information — Infographie et traitement de l'image — Spécification fonctionnelle pour le traitement de l'image et l'échange (IPI) —

Partie 5: Format d'échange de l'image de base (BIIF)

Reference number
ISO/IEC 12087-5:1998(E)

Contents

Contents ii
List of Figures v
List of Tables vi
Foreword viii
1 Scope 1
2 Normative references 2
3 Terms, definitions and abbreviations 4
3.1 Definitions 4
3.2 Abbreviations: 7
4 Basic Image Interchange Format (BIIF) specification 8
4.1 Format overview 8
4.1.1 Description 9
4.1.1.1 Format fields 9
4.1.1.2 Standard data types 9
4.1.1.3 Extensions 9
4.1.1.4 Transportable file structure (TFS) 10
4.1.1.5 Complexity Levels 10
4.1.2 Interoperability/exchange 10
4.1.3 Fields 11
4.1.3.1 Valid data 11
4.1.3.2 Date and time expressions 11
4.1.3.3 Representation of textual information in fields 11
4.1.4 Logical structure of pixel storage 12
4.1.4.1 Pixel coordinates 12
4.1.4.2 PIKS pixel coordinates 12
4.1.5 Common coordinate system 13
4.1.5.1 Common coordinate system structure 13
4.1.5.2 Row and column coordinates 13
4.1.5.3 CCS Boundaries 14
4.1.6 Display and attachment levels 14
4.1.6.1 Display levels 14
4.1.6.2 Attachment levels 15
4.2 Format 16
4.2.1 Data recording formats 16
4.2.2 Encoding 17
4.2.3 Header 18
4.2.4 Image segment 26
4.2.4.1 Image subheader 26
4.2.4.2 Look-up tables (LUTS) 27
4.2.5 Image data field format 36
4.2.5.1 Blocked images 36
4.2.5.2 Image data masking 37
4.2.5.3 Compressed image data format 42
4.2.5.4 Uncompressed image data format 42
4.2.5.5 Vector quantized data 43
4.2.5.6 Number of bands (NBANDS) 43
4.2.5.7 PIKS Objects 43
4.2.5.8 Image representation 44

4.2.6 Symbol segment........ 45
4.2.6.1 Symbol subheader........ 45
4.2.6.2 Symbol data........ 48
4.2.7 Text information segment........ 48
4.2.7.1 Text subheader........ 48
4.2.7.2 Text........ 50
4.2.8 Data Extensions........ 50
4.2.8.1 Tagged Record Extensions (TRE): Public and Private......... 50
4.2.8.2 Data extension segments (DES)........ 52
4.2.8.3 Defined DESs........ 54
4.2.8.4 Reserved extension segments (RES)........ 56

5 Conformance profiles and extensions........ 58
5.1 Profiles........ 58
5.2 Profile specific header/subheader dependencies proforma........ 58
5.3 Complexity level proforma........ 58
5.4 Implementation support requirements........ 59
5.4.1 General support requirements........ 59
5.4.2 Producing and interpreting BIIF files........ 59
5.4.2.1 Producing BIIF files........ 59
5.4.2.2 Interpreting BIIF files........ 59
5.5 Defined extensions........ 59
5.6 Profile registration........ 59

Annex A (normative) Transportable file structure........ 61
A.1 Transportable file structure (TFS)........ 61
A.1.1 TFS Commands........ 61
A.1.2 TFS encoding........ 62
A.1.3 TFS command flow........ 62
A.2 TFS command formats........ 63
A.2.1 TFS delimiter commands........ 63
A.2.1.1 Begin TFS command........ 63
A.2.1.2 Begin transport command........ 64
A.2.1.3 Begin transport body command........ 64
A.2.1.4 Begin transport profile command........ 64
A.2.1.5 Begin transport profile body command........ 65
A.2.1.6 End transport profile command........ 65
A.2.1.7 End transport command........ 65
A.2.1.8 End TFS command........ 65
A.2.2 TFS descriptor commands........ 65
A.2.2.1 TFS version command........ 66
A.2.2.2 TFS security command........ 66
A.2.2.3 TFS subscription command........ 66
A.2.2.4 TFS configuration command........ 67
A.2.2.5 TFS configuration data command........ 68
A.2.2.6 TFS metadata command........ 68
A.2.2.7 TFS index command........ 68
A.2.3 Transport descriptor commands........ 69
A.2.3.1 Transport security command........ 69
A.2.3.2 Transport metadata command........ 70
A.2.3.3 Transport index command........ 70
A.2.4 Transport Profile descriptor commands........ 70
A.2.4.1 Transport Profile security command........ 70
A.2.4.2 Transport profile metadata command........ 70
A.2.4.3 Transport profile index command........ 70
A.2.5 Transport profile commands........ 70
A.2.5.1 Transport profile action command........ 71
A.2.5.2 Transport profile object command........ 71
A.2.6 TFS escape command........ 79

Annex B (normative) Vector Quantization........ 80
B.1 Vector Quantized Data........ 80
B.2 Quantization Process........ 80
B.3 Reconstruction........ 81
B.3.1 Spatial reconstruction........ 82
B.3.2 Color reconstruction........ 83
B.3.3 Data elements........ 84

B.3.3.1 Quantization ratio 84
B.3.3.2 Masked vs unmasked 84
B.3.3.3 Code book organization 84
B.3.3.4 Spatial data section 85
B.4 File organization 85
B.5 Definitions - image data section 87
B.6 Definitions - BIIF header and image subheader 87

Annex C (normative) Profiling BIIF 89
C.1 Profiling process 89
C.2 Profile proforma 89
C.2.1 Use of the model profile 89
C.2.2 Rules for filling out the proforma tables 89
C.2.2.1 Profile Tables 91

Annex D (informative) Implementation Considerations and Product Configurations 123
D.1 Implementation considerations and product configurations 123
D.2 TRE_OVERFLOW example 123
D.3 Scope of implementation 123
D.3.1 Creating headers and subheaders 123
D.3.2 Character counts 123
D.3.3 Data entry 124
D.3.4 Tagged Record Extensions 124
D.3.5 Out-of-bounds field values 124
D.3.6 Use of images 124
D.3.7 Use of text files 124
D.3.7.1 BCS (TXTFMT=STA) 124
D.3.7.2 Additional TXTFMT Codes 124
D.3.8 File system constraints 130
D.3.9 Security considerations 130
D.4 Product configurations 131
D.4.1 General 131
D.4.1.1 Single file, single base image 131
D.4.1.2 Single file, multiple images 132
D.4.1.3 Single file, no image 134
D.4.1.4 Multiple correlated files 134

Annex E (informative) Examples BIIF Profiles 135
E.1 Example file 135
E.1.1 Use of this example 135
E.1.2 BIIF Model Profile for example file 135
E.1.2.1 Explanation of the file header 137
E.1.2.2 Explanation of the image subheaders 138
E.1.2.3 Explanation of the symbol subheaders 142
E.1.2.4 Explanation of the text subheaders 147
E.2 Examples using the Transportable File Structure (TFS) 147
E.2.1 Use of TFS 147
E.2.2 TFS Examples 147
E.2.2.1 TFS example of PIKS processing for an image 147
E.2.2.2 TFS example of requesting imagery and patient history 148
E.2.2.3 TFS example of using configuration and a five band image 148
E.3 Open Skies Digital Data Exchange Profile example (Informative) 149
E.3.1 Open Skies Digital Data Exchange File Header 150
E.3.2 Open Skies Image Data Subheader 151
E.3.3 Open Skies SAR Information 154
E.3.4 Open Skies Text Data Subheader 156
E.3.5 Open Skies Annotation Text Format 157
E.3.6 Data Extension Segment Subheader 159
E.3.7 Open Skies Digital Data Exchange Format (OSDDEF) DCRsi TAPE ANNOTATION and TAPE DIRECTORY File Examples 160
E.3.8 Example Entries in Data and Field Tables 163

List of Figures

Figure 1 -- Translation process 8
Figure 2 -- Structure 9
Figure 3 -- Image Array Structure 12
Figure 4 -- Common coordinate system 13
Figure 5 -- Display level and attachment level relationships 15
Figure 6 -- Display and attachment levels 16
Figure 7 -- Octet sequence order 17
Figure 8 -- Header structure 18
Figure 9a -- A blocked image 37
Figure 9b -- A blocked padded image 37
Figure 10 -- A blocked padded image with empty blocks 38
Figure B.1 -- Vector quantization process flow 80
Figure B.2 -- BIIF file structure with VQ data 81
Figure B.3 -- VQ reconstruction procedure 82
Figure B.4 -- Spatial reconstruction 83
Figure B.5 -- Color reconstruction 84
Figure B.6 -- Structure of the BIIF VQ image data section 86
Figure D.1 -- Single file, single base image representation 131
Figure D.2 -- Single file, multiple images representation 133
Figure E.1 -- Sample file composite image 135

List of Tables

Table 1 -- Header ... 19
Table 2 -- Image pixel data type codes and descriptions ... 27
Table 3 -- Image subheader ... 28
Table 4 -- Image data mask table ... 39
Table 5 -- Representative examples of IREP and associated fields ... 45
Table 6 -- Symbol subheader ... 46
Table 7 -- Text subheader ... 49
Table 8 -- TRE subheader ... 52
Table 9 -- Data extension segment subheader ... 53
Table 10 -- Data extension segment subheader for TRE OVERFLOW ... 54
Table 11 -- Data extension segment subheader for TRANSPORTABLE_FILE_STRUCT ... 55
Table 12 -- Reserved extension segment subheader ... 57
Table A.1 -- Encoding of a TFS command ... 62
Table A.2 -- TFS delimiter command table ... 63
Table A.3 -- TFS command even form ... 64
Table A.4 -- TFS command odd form ... 64
Table A.5 -- TFS descriptor command table ... 65
Table A.6 -- TFS version ... 66
Table A.7 -- TFS subscription command ... 67
Table A.8 -- TFS subscription contents table ... 67
Table A.9 -- TFS configuration command ... 68
Table A.10 -- TFS config data command ... 68
Table A.11 -- TFS index command ... 69
Table A.12 -- Transport descriptor command table ... 69
Table A.13 -- Transport profile descriptor command table ... 70
Table A.14 -- Transport profile command table ... 71
Table A.15 -- Transport profile object command ... 72
Table A.16 -- Object data for BIIF transport profile object ... 72
Table A.17 -- Object storage and object data format ... 73
Table A.18 – PIKS Objects minimum size ... 73
Table A.19 -- PIKS objects data structures ... 74
Table B.1 -- BIIF header and subheader specified data values ... 88
Table C.1 -- File header fields ... 91
Table C.2 -- Security fields specification ... 96
Table C.3 -- Image subheader fields ... 97
Table C.4 -- Image data mask table ... 107
Table C.5 -- Symbol subheader ... 108
Table C.6 -- Text subheader ... 111
Table C.7 -- Tagged record extensions ... 112
Table C.8a -- Data extension segment proforma ... 113
Table C.8b -- Reserved extension segment proforma ... 114
Table C.9 -- TFS profile proforma table ... 115
Table C.10 – Implementation Support Requirements ... 122
Table D.1 -- Basic Latin character set ... 125
Table D.2 -- Basic Latin character set explanation ... 126
Table D.3 -- Latin-1 supplement character set ... 128
Table D.4 -- Latin-1 supplement character set explanation ... 129
Table E.1 -- Example BIIF file header ... 136
Table E.2 -- Example of the first image subheader ... 138
Table E.3 -- Example of the second image subheader ... 140
Table E.4 -- Symbol subheader for the first symbol ... 142
Table E.5 -- Symbol subheader for the second symbol ... 143
Table E.6 -- Symbol subheader for the third symbol ... 144
Table E.7 -- Symbol subheader for the fourth symbol. ... 145
Table E.8 -- Symbol subheader for the fifth symbol ... 146
Table E.9 -- Text subheader for the text document ... 147
Table E.10 -- Open Skies Digital Data Exchange File Header ... 150
Table E.11 -- Image Data Subheader ... 151
Table E.12 -- Tagged Record Extension for SAR Information Parameters ... 154
Table E.13 -- SAR Information Parameters ... 154

Table E.14 -- Text Data Subheader 156
Table E.15 -- Treaty on Open Skies Annotation Text Format 157
Table E.16 -- Data Extension Segment Subheader 159
Table E.17 -- Tape annotation file example 160
Table E.18 -- Tape annotation file example 160
Table E.19 -- Tape annotation file 161
Table E.20 -- Tape Directory File Example 161
Table E.21 -- Tape Directory File 162
Table E.22 -- Data Field Table Example 162
Table E.23 -- Header example for various exchange files examples 163
Table E.24 -- Image data subheader 164
Table E.25 -- Text subheader 165
Table E.26 -- RETAG data for SAR initial phase data 165
Table E.27 -- User defined registered extension 166
Table E.28 -- Expansion of SAR micronavigational parameters 166
Table E.29 -- Configuration of SAR 166
Table E.30 -- 2nd example of configuration of SAR 166
Table E.31 -- Data extension segment for SAR 167

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

International Standard ISO/IEC 12087-5 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 24, *Computer graphics and image processing*.

ISO/IEC 12087 consists of the following parts, under the general title *Information technology — Computer graphics and image processing — Image Processing and Interchange (IPI) — Functional specification*:

– *Part 1: Common architecture for imaging*

– *Part 2: Programmer's imaging kernel system application programme interface*

– *Part 3: Image Interchange Facility (IIF)*

– *Part 4: PICS — DTC*

– *Part 5: Basic Image Interchange Format (BIIF)*

Annexes A to C form an integral part of this part of ISO/IEC 12087. Annexes D and E are for information only.

Information technology — Computer graphics and image processing — Image Processing and Interchange (IPI) — Functional Specification —

Part 5:
Basic Image Interchange Format (BIIF)

1 Scope

This part of ISO/IEC 12087 establishes the specification of the Basic Image Interchange Format (BIIF) part of the standard. BIIF is a standard developed to provide a foundation for interoperability in the interchange of imagery and imagery-related data among applications. This part of ISO/IEC 12087 provides a detailed description of the overall structure of the format, as well as specification of the valid data and format for all fields defined with BIIF. Annex C contains a model profile in tables to assist in profile development.

As part of the ISO/IEC 12087 family of image processing and interchange standards, BIIF conforms to the architectural and data object specifications of ISO/IEC 12087-1, the Common Architecture for Imaging. BIIF supports a profiling scheme that is a combination of the approaches taken for ISO/IEC 12087-2 (PIKS), ISO/IEC 10918 (JPEG), ISO/IEC 8632 (CGM), and ISO/IEC 9973 (The Procedures for Registration of Graphical Items). It is intended that profiles of the BIIF will be established as an International Standardised Profile (ISP) through the normal ISO processes (ISO/IEC TR 10000).

The scope and field of application of this part of ISO/IEC 12087 includes the capability to perpetuate a proven interchange capability in support of commercial and government imagery, Programmer's Imaging Kernel System Data, and other imagery technology domains in that priority order.

This part of ISO/IEC 12087 provides a data format container for image, symbol, and text, along with a mechanism for including image-related support data.

This part of ISO/IEC 12087 satisfies the following requirements:

- Provides a means whereby diverse applications can share imagery and associated information.
- Allows an application to exchange comprehensive information to users with diverse needs or capabilities, allowing each user to select only those data items that correspond to their needs and capabilities.
- Minimizes preprocessing and postprocessing of data.
- Minimizes formatting overhead, particularly for those applications exchanging only a small amount of data and for bandwidth-limited systems.
- Provides a mechanism (Transportable File Structure, TFS) to interchange PIKS image and image-related objects
- Provides extensibility to accommodate future data, including objects.

When the extensibility of this part of ISO/IEC 12087, or the inherent constraints of the structured format of BIIF, do not meet the needs of a more complex application, the concepts and features of 12087-3 (IIF) should be considered as a more appropriate method of image interchange. For example, the ability to support complex combinations of heterogeneous pixel types, self defining pixel structures, or abstract structures can be done with IIF.

2 Normative references

The following standards contain provisions which, through reference in this text, constitute provisions of this part of ISO/IEC 12087. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this part of ISO/IEC 12087 are encouraged to investigate the possibility of applying the most recent editions of the standards indicated below. Members of IEC and ISO maintain registers of currently valid International Standards.

ISO/IEC 646:1991, *Information technology - ISO 7-bit coded character set for information interchange.*

ISO/IEC 8601:1988/Cor.1:1991, *Data elements and interchange formats - Information interchange - Representation of dates and times - Technical Corrigendum 1.*

ISO/IEC 8632-1:1992, *Information technology - Computer graphics - Metafile for the storage and transfer of picture description information - Part 1: Functional specification.*

ISO/IEC 8632-1:1992/Amd.1:1994, *Information technology - Computer graphics - Metafile for the storage and transfer of picture description information - Part 1: Functional specification - Amendment 1: Rules for profiles.*

ISO/IEC 8632-1:1992/Amd.2:1995, *Information technology - Computer graphics - Metafile for the storage and transfer of picture description information - Part 1: Functional specification - Amendment 2: Application structuring extensions.*

ISO/IEC 8632-2:1992, *Information technology - Computer graphics - Metafile for the storage and transfer of picture description information - Part 2: Character encoding.*

ISO/IEC 8632-3:1992, *Information technology - Computer graphics - Metafile for the storage and transfer of picture description information - Part 3: Binary encoding.*

ISO/IEC 8632-4:1992, *Information technology - Computer graphics - Metafile for the storage and transfer of picture description information - Part 4: Clear text coding.*

ISO/IEC 8632-4:1992/Amd.1:1994, *Information technology - Computer graphics - Metafile for the storage and transfer of picture description information - Part 4: Clear text coding - Amendment 1: Metafile for the storage and transfer of picture description information.*

ISO/IEC 8632-4:1992/Amd.2:1995, *Information technology - Computer graphics - Metafile for the storage and transfer of picture description information - Part 4: Clear text coding - Amendment 2: Application structuring extensions.*

ISO/IEC 9973:1994, *Information technology - Computer graphics and image processing - Procedures for registration of graphical items.*

ISO/IEC TR 10000-1:1995, *Information technology - Framework and taxonomy of International Standardized Profiles - Part 1: General principles and documentation framework.*

ISO/IEC 10646-1:1993, *Information technology - Universal Multiple-Octet Coded Character Set (UCS) - Part 1 : Architecture and Basic Multilingual Plane.*

ISO/IEC 10646-1:1993/Amd.2:1996, *Information technology - Universal Multiple-Octet Coded Character Set (UCS) - Part 1: Architecture and Basic Multilingual Plane - Amendment 2: UCS Transformation Format 8 (UTF-8).*

ISO/IEC 10918-1:1994, *Information technology - Digital compression and coding of continuous-tone still images: Requirements and guidelines.*

ISO/IEC 10918-2:1995, *Information technology - Digital compression and coding of continuous-tone still images: Compliance testing.*

ISO/IEC 10918-3:1997, *Information technology - Digital compression and coding of continuous-tone still images: Extensions.*

ISO/IEC 10918-4:—[1], *Information technology - Digital compression and coding of continuous-tone still images: Registration of JPEG profiles, SPIFF profiles, SPIFF tags, SPIFF colour spaces, APPN markers, SPIFF compression types and Registration Authorities (REGAUT).*

[1] To be published.

ISO/IEC 12087-1:1995, *Information technology - Computer graphics and image processing - Image Processing and Interchange (IPI)-- Functional specification - Part 1: Common architecture for imaging.*

ISO/IEC 12087-2:1994, *Information technology - Computer graphics and image processing - Image Processing and Interchange (IPI) --Functional specification - Part 2: Programmer's imaging kernel system application programme interface.*

ISO/IEC 12087-3:1995, *Information technology - Computer graphics and image processing - Image Processing and Interchange (IPI) -- Functional specification - Part 3: Image Interchange Facility (IIF).*

ANSI/IEEE std 754:1985, *Standard for Binary Floating-Point Arithmetic.*

ITU-T T.4 (1993:03)/Amd.2:1995, *Standardisation of Group 3 Facsimile apparatus for document transmission*

3 Terms, definitions and abbreviations

3.1 Definitions

For the purposes of this part of ISO/IEC 12087, the following definitions apply.

3.1.1
Attachment Level
A field value of a segment that indicates the display level of the segment to which it is attached. It provides a way to associate images and symbols as a group for the purpose of moving, rotating or displaying.

3.1.2
Annotational Text
See Symbol Text.

3.1.3
Band
One of the two-dimensional (row/column) arrays of pixel sample values that comprise an image. For the basic use of BIIF, the band values are homogeneous data types for each band. In the case of monochrome or indexed colour images (single 2 dimensional array of pixel values with possible look-up-tables), the image array consists of one band. In the case of RGB images (three 2-dimensional arrays of pixel values; 8 bits each of Red, Green and Blue values for each pixel), the image consists of three bands. When images need to be represented using bands with heterogeneous array structure or data types (e.g., two bands with integer data type and one band with a real data type), the image representation may be defined using a PIKS object in a TFS Data Extension Segment (DES). The TFS PIKS object defines the data structure of the values in the image data field of the image segment.

3.1.4
Basic Character Set (BCS)
A subset of ISO/IEC 10646-1 character set which is represented by the UTF-8 form and used in headers and subheaders.

3.1.5
Basic Character Set-Alphanumeric (BCS-A)
A subset of the Basic Character Set. The range of allowable characters consists of space through tilde (single octets with values ranging from 20 to 7E) from the Basic Latin Collection.

3.1.6
Basic Character Set-Numeric (BCS-N)
A subset of the Basic Character Set which consists of the digits ‘0’ through ‘9’, ‘plus sign’, ‘minus sign’, ‘decimal point’, and ‘slash’.

3.1.7
Basic Multilingual Plane
The Basic Multilingual Plane (BMP) is defined as group 00 of plane 00. The BMP includes characters in general use in alphabetic, syllabic, and ideographic scripts together with various symbols and digits. See ISO/IEC 10646-1:1993.

3.1.8
Block
A rectangular array of pixel values which is a subset of an image. An image consists of the union of one or more non-overlapping blocks.

3.1.9
Byte
A byte is defined as equivalent to an octet.

3.1.10
Common Coordinate System
A two dimensional coordinate space which is common for determining the placement and orientation of displayable data types within a specific BIIF file and among correlated BIIF files which comprise an integrated product.

3.1.11
Conditional
An adjective applied to data fields whose existence depends on the value of the designated Required field preceding the Conditional field.

3.1.12
Coordinated Universal Time (UTC)
The time scale maintained by the Bureau International de L'Heure (International Time Bureau) that forms the basis of a coordinated dissemination of standard frequencies and time signals. UTC is equivalent to the mean solar time at the prime meridian at Greenwich, England.

3.1.13
DES
Data Extension Segment is a construct used to encapsulate different data types where each type is encapsulated in its own DES. The DES structure is discussed in Subclause 4.2.8.2.

3.1.14
Displayable
Information that can be exhibited in visual form.

3.1.15
Display Level
A field value of a segment that denotes the order in which the segments (images and symbols) are "stacked". The Display Level order is independent of the data sequence order in this format.

3.1.16
Field
Logically primitive item of data, sometimes referred to as an attribute.

3.1.17
Image
A representation of physical visualization, for example, a picture. An image is the computer (digital) representation of a picture. An image is comprised of discrete picture elements called pixels structured in an orderly fashion consisting of pixel value arrays formatted using bands and blocks.

3.1.18
International Standardized Profile
An internationally agreed-to, harmonized document which identifies a standard or group of standards together with options and parameters necessary to accomplish a function or set of functions.

3.1.19
Look-Up Table
A collection of values used for translating image samples from one value to another. The current sample value is used as an index into the look-up table(s); therefore, the number of entries in each look-up table for a single bit per pixel image would contain two entries, and each look-up table for an 8-bit per pixel image would contain 256 entries. Multiple look-up tables allow for the translation of a scalar pixel value to an n-dimensional vector pixel value.

3.1.20
Non-blank
Non-blank indicates that the field cannot be filled entirely by the BCS-A space character (0x20). It may contain space characters when included with other characters.

3.1.21
Octet
An octet is defined as 8 bits.

3.1.22
Pad Pixel
A pixel with sample values that have no significant meaning to the image. Pad pixels are used with block images when either the number of pixel rows in an image is not an integer multiple of the desired number of vertical image blocks, or when the number of pixel columns in an image is not an integer multiple of the desired number of horizontal image blocks.

3.1.23
Pad Pixel Mask
A data structure which identifies recorded/transmitted image blocks which contain pad pixels. The pad pixel mask allows applications to identify image blocks which require special interpretation due to pad pixel content.

3.1.24
Pixel
An abbreviation for the term "picture element".

3.1.25
Profile
A set of one or more base standards, and where applicable, the identification of chosen classes, subsets, options, and parameters of those base standards, necessary for accomplishing a particular function.

3.1.26
Profile Variant
A field within this basic standard that is allowed to be defined by a profile for its structure and intent (content). An element or an attribute that is allowed to differ between profiles.

3.1.27
Required
An adjective applied to data fields that must be present and filled with valid data or default data.

3.1.28
RES
The Reserved Extension Segment (RES) construct provides the same mechanism as the DES construct for adding new data types for inclusion in BIIF files. However, the RES is reserved for data types that need to be placed at or near the end of the file. The RES structure is discussed in Subclause 4.2.8.4.

3.1.29
Sample
One element in the image array that comprises an attribute of the image. In BIIF, a sample (pixel vector value) is indexed according to the row and column of the array where it appears.

3.1.30
SAR
Image obtained from a synthetic aperture radar.

3.1.31
SARIQ
Radio hologram (initial phase information) from a synthetic aperture radar.

3.1.32
Segment
An instance of a data type that is contained in a BIIF file. A segment is comprised of a subheader and associated data (e. g., an image subheader together with image data comprises an image segment).

3.1.33
Symbol
A pictorial element that may be aligned with a point in or adjacent to an image to provide graphical markings and/or textual labels.

3.1.34
Symbol Text
Text placed on or adjacent to an image as a graphic symbol to provide a textual overlay to the image.

3.1.35
Tagged Record Extension (TRE)
A means to provide additional attributes about standard data segments not contained in the standard header or sub-header fields.

3.1.36
Transportable File Structure
Transportable File Structure is a data extension element used for configuration, data request, commands, and PIKS object data to be stored in hierarchy order with Metadata associated for each level. TFSs are defined and demonstrated in Annex A.

3.1.37
Transparent Pixel
A pixel whose sample values must be interpreted for display such that the pixel does not obscure the display of any underlying pixel.

3.1.38
YCbCr
Technique for specifying colour images where Y = Brightness of signal, Cb = Chrominance (blue), Cr = Chrominance (red)

3.2 Abbreviations:

AL	Attachment Level
BCS	Basic Character Set
BCS-A	Basic Character Set - Alphanumeric
BCS-N	Basic Character Set - Numeric
BMP	Basic Multilingual Plane
C	Conditional
CCS	Common Coordinate System
CGM	Computer Graphics Metafile
CS	Character String
DES	Data Extension Segments(s)
DL	Display Level
JPEG	Joint Photographic Experts Group
LSB	Least Significant Bit
LUT	Look-Up Table
MSB	Most Significant Bit
NBPC	Number of Blocks Per Column
NBPR	Number of Blocks Per Row
PIKS	Programmer's Imaging Kernel System
PIKS1	PIKS Foundation Profile
PIKS2	PIKS Technical Profile
PIKS3	PIKS Scientific Profile
PIKS4	PIKS Full Profile
PVTYPE	Pixel Value Type
RES	Reserved Extension Segment (s)
RGB	Red, Green, Blue
YCbCr	Y-Brightness of signal, Cb-Chrominance (blue), Cr-Chrominance (red)
TRE	Tagged Record Extension
TFS	Transportable File Structure
UCS	Universal Multi-Octet Coded Character Set
UCS-2	UCS Two Octet Form
UCS-4	UCS Four Octet Form
UTC	Coordinated Universal Time
UTF-8	UCS Transformation Format 8

4 Basic Image Interchange Format (BIIF) specification

4.1 Format overview

Imagery applications use multiple types of systems for the exchange, storage, and processing of images and associated imagery data. The format used in one application is likely to be incompatible with formats used in other applications. Since each application may use a unique, internal data representation, a common format for interchange of information across applications is needed for interoperability of systems within and across applications. This clause defines the Basic Image Interchange Format (BIIF) specification to provide a common basis for storage and interchange of images and associated data among existing and future applications. BIIF supports interoperability by providing a data format for shared imagery and an interchange format for images and associated imagery data.

In BIIF, data interchange between disparate systems is potentially enabled by a translation process. Using BIIF, each system must be compliant with only one external format that will be used for communication with all other participating systems. When BIIF is not used as a system's native internal format, each system will translate between the system's internal representation for imagery and the BIIF format. A system from which data is to be transferred has a translation module that accepts information structured according to the system's internal representation for images and related imagery data, and assembles this information into BIIF format. The receiving system will reformat BIIF data, converting it into one or more files structured as required by the internal representation of the receiving system. Each receiving system can translate selectively and permanently store only those portions of data in the received BIIF file that are of interest. A system may transmit all of its data, even though some of the receiving systems may be unable to process certain elements of the data. The functional architecture of this translation process is shown in Figure 1. In the diagram, the terms "Native Mode" refer to imagery and imagery-related data represented in a way potentially unique to the sending or receiving system, respectively.

Flexibility and extensibility of the use of BIIF are provided through the use of a constrained set of conditional variable length fields and extension constructs. Imagery, together with Programmer's Imaging Kernel System (PIKS) objects, can easily be formatted into BIIF. This approach provides the proven capability to implement general purpose BIIF readers (applications) that can present the basic imagery and annotations of any BIIF compliant product file created within the constraints of a given profile of BIIF. Although more robust approaches exist to allow 'self-defining' data structures, these approaches significantly increase the complexity for implementing general purpose readers (applications) capable of meaningful interpretation of file constructs created by a wide variety of diversely developed generators. More simplistic imagery file formats also exist. They are often focused at just portraying a simple digital image and are often too limited in feature sets to meet the needs of somewhat more sophisticated, but still basic imagery applications. BIIF provides a basic capability that bridges the gap between simplistic digital image formats and the extremely sophisticated, self-defining, but potentially complex formats. As such, BIIF has some inherent bounds and limitations, but remains very capable as a basic imagery format capable of satisfying a broad range of imagery applications.

Although few in number, certain aspects of this format reflect the legacy formats it is based on. For example, pixel coordinate indexing as described in 4.1.4.2.

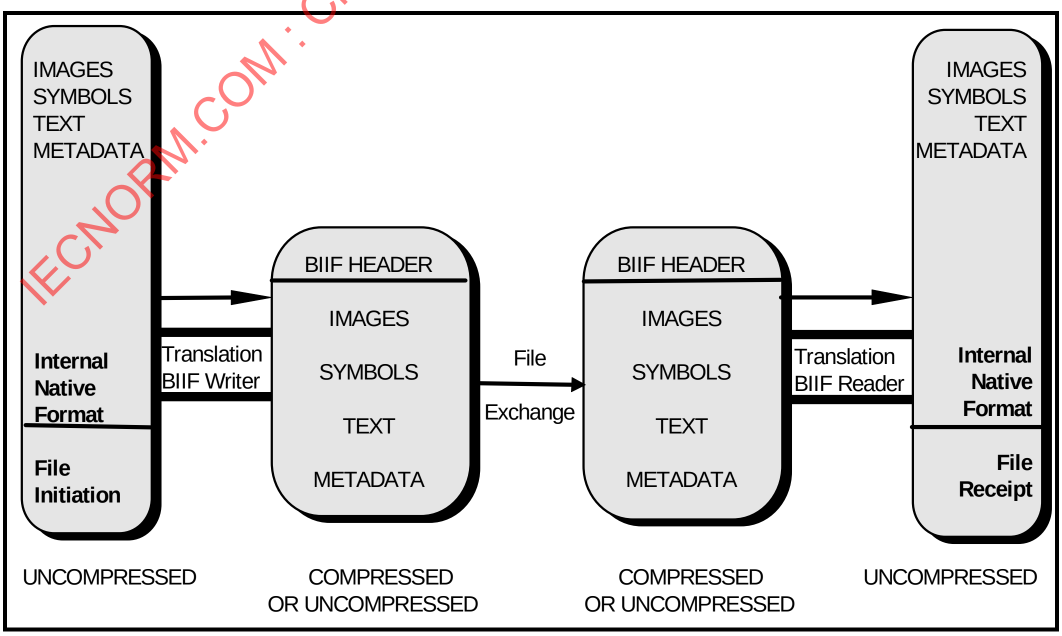


Figure 1 -- Translation process

BIIF supports extensibility (through the use of its Data Extension Segment (DES) and Reserved Extension Segment (RES)) which empowers new applications while maintaining backward compatibility. Newly defined data can be linked via a DES/RES and utilized by a new application or ignored by a legacy system. This will insure backward compatibility to older systems while incorporating new technology.

4.1.1 Description

4.1.1.1 Format fields

BIIF format structure consists of a combination of both fixed octet length fields and variable octet length fields. In its most basic application, the use of variable length fields is minimized to deal with the basic issues of the variable number and size of images and the variable number and size of image annotations that may appear in a BIIF file. The format contains header, subheader, and data fields. BIIF header and subheader fields are octet aligned. The file header carries information about the identification, security, structure, content, size of the file as a whole, and size of the data segments within the file. A data segment structure is defined for each kind of data supported by the format. Each data segment in the file has a subheader containing information that describes characteristics of the data segment and an associated data field that contains the actual data. BIIF encoding is discussed in Subclause 4.2.3.

4.1.1.2 Standard data types

A BIIF file supports inclusion of three standard types of data in a single file: image, symbol, and text. It is possible to include zero, one, or multiple data segments of each standard data type in a single file (for example: several images, but no symbols). Standard data types shall be placed in the file in the following order: all image data segments (images), followed by all symbol data segments (symbols including symbol text), followed by all text segments (documents or text). Additional kinds of data may be included in a BIIF file by use of Data Extension Segments (DES) (see Subclause 4.2.8), such as the Transportable File Structure (TFS Annex A), and Reserved Extension Segments (RES) (see Subclause 4.2.9). A data segment of a standard data type is called a standard data segment. A data segment of a type defined in a DES or RES is called an extension data segment. The order of these major file components is illustrated in Figure 2.

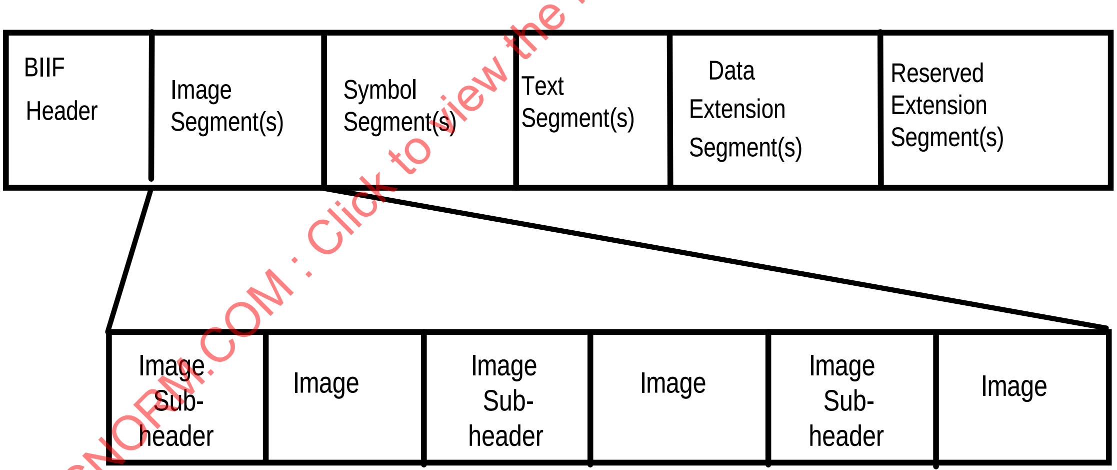


Figure 2 -- Structure

4.1.1.3 Extensions

Flexibility to add support for kinds of data and data characteristics not explicitly defined in this standard is provided within the format. This is accomplished by providing for one or two fields in each header/subheader containing "tagged record extensions" and by use of DESs and RESs. The tagged records extensions (TREs) in the header/subheaders may contain additional characteristics about the corresponding data segment, while the extension segments are intended primarily to provide a vehicle for adding support for new kinds of data. The identifier (tag name) for the tagged records, and extension segment identifiers, will be coordinated centrally in accordance with this standard to avoid conflicting use. In some cases, tagged record and extension formats will be configuration managed to control changes to data formats affecting a broad BIIF user base.

All implementations of BIIF should handle the receipt of unknown extension types by at least recognizing they are unknown extension types and ignoring them. This is accomplished through the octet count mechanism of the extension identifier plus length field approach for bounding separate content types in the format. The octet length offsets allow an interpret implementation to skip past unknown extensions and interpret the known meaningful content elements of the BIIF file. This concept is basic to BIIF interpretation; a reader can always identify octet count offsets and move on to other elements of the file. A quality implementation will have the provision to alert the user of unknown entities that are skipped over to ensure the user does not infer from the presentation of the file that all file contents have been fully interpreted and made available for use by the user.

The BIIF file header and each standard data segment subheader have designated expandable fields to allow for the optional inclusion of extension data (Tagged Record Extensions). The inclusion of extension data provides the ability to add data/information about the standard data segment (metadata) that is not contained in the basic fields of the headers and subheaders. The additional data is contained within one or more BIIF Tagged Record Extensions that are placed in the appropriate field (user defined data field or extended data field) of the standard data segment subheader for which the metadata applies. When Tagged Record Extensions have application across multiple data segments in the file, or otherwise apply to the entire BIIF file in general, they are placed in the appropriate file header extension fields.

Exemplary use of Tagged Record Extensions:

- Data about people, buildings, places, landmarks, equipment or other objects that may appear in the image.
- Data to allow correlation of information among multiple images and annotations within a BIIF file.
- Data about the equipment settings used to obtain the digital image, xray, etc.
- Data to allow geopositioning of items in the imagery or measurement of distances of items in the imagery.

4.1.1.4 Transportable file structure (TFS)

The Transportable File Structure (TFS) Data Extension Element allows for the transport of the Programmer's Imaging Kernel System (PIKS) object data and other image related data. The TFS is a simple metafile that allows for both relational and hierarchical structures of image-related data to be stored in BIIF. Each TFS contains one or more Transports. The Transports allow for the grouping of image-related data for a specific reason. For example, the Transport could contain all the required information about a medical patient to include imagery, PIKS objects for the imagery, and a complete medical history of the patient. In the case of remote sensing, the Transport could contain area of interests, specific locations, and explanations of features on the image. Each Transport contains one or more Profiles which contain PIKS object data or other image-related object data. TFS object data either contains data for the object or provides an unambiguous reference to the data. Profiles are related together when they occur at the same level in the metafile. For example, when a Transport contains an image Profile and several PIKS object Profiles, it is implied that the PIKS objects are related to the image. A Profile can also be nested within a Profile to represent complex hierarchical structures. Each TFS, Transport, and Profile contains metadata about itself and its contents, security about the data, and an index into the next levels for quick access. Although the TFS can be used to describe complex data structures about the image, PIKS data objects relating to an image can be simply expressed using one Transport and one Profile. The Transportable File Structure is completely described in Annex A. Example uses of TFS are given in Annex E.

4.1.1.5 Complexity Levels

The BIIF uses the concept of complexity levels to allow definitions of a set of nested features and constraints to exist within a user domain served by a specific profile of BIIF. It allows a user application to quickly identify the degree of complexity of the number and combinations of BIIF features used in any specific BIIF file within this given profile. For Example, the BIIF Model Profile in Annex C allows a file marked at complexity level 01 to have one image segment and no symbol segments, text segments, or data extension segments. A Model Profile file marked at complexity level 02 may have up to 20 image segments, 100 symbol segments, 10 text segments and 20 data extension segments.

4.1.2 Interoperability/exchange

Within a BIIF Profile, the term interoperability is used to express the ability of two or more imagery users within the same community of interest operating in a heterogeneous computing environment, to accurately create and/or recognize BIIF file structure as prescribed in this standard, and meaningfully exchange the information contained in it. This standard also promotes a higher degree of interoperability among two or more diverse communities of interest through the selection of a common set of functionality captured in specific profiles and implementation agreements.

4.1.3 Fields

4.1.3.1 Valid data

All header and subheader fields contained in the BIIF file shall contain valid data (that is, data in accordance with the restrictions specified for the contents of the field in this document or as constrained by an applicable profile definition; Annex C) or the specified default value. In this standard the word ‘profile’ shall refer to a registered BIIF profile that describes the field entries for each subclause of BIIF.

4.1.3.2 Date and time expressions

The representation of date and time shall be of the form CCYYMMDDhhmmss in accordance with the provisions of ISO 8601:1988(E) for expressing combined date and time of day representations with the century designator. All dates and times shall be expressed in Coordinated Universal Time (UTC).

The associated meaning of the CCYYMMDDhhmmss representation is:

— [CC] represents the digits used in the thousands and hundreds components of the time element “year”. The range of ‘CC’ is ‘00’ through ‘99’.
— [YY] represents the digits used in the tens and units components of the time element "year". The range of 'YY' is '00' through '99'.
— [MM] represents the digits used in the time element "month". The range of 'MM' is '01' through '12'.
— [DD] represents the digits used in the time element "day". The range of 'DD' is '01' through '31'.
— [hh] represents the digits used in the time element "hour". The range of 'hh' is '00' through '23'.
— [mm] represents the digits used in the time element "minute". The range of 'mm' is '00' through '59'.
— [ss] represents the digits used in the time element "second". The range of 'ss' is '00' through '59'.

4.1.3.3 Representation of textual information in fields

BIIF uses two different categories (BCS, UCS) of textual data character representations. Each category has a set of constraints for use within header and subheader fields. UTF 8 is used in selected header fields. BCS (BCS-A and BCS-N) and UCS (UTF 8) character representation are allowed in header and sub-header fields. BCS and UCS character codes are allowed for use in the data fields of text segments (see 4.1.3.3.1 to 4.1.3.3.5).

4.1.3.3.1 Basic character set

The Basic Character Set (BCS) is used when populating header and subheader fields of BIIF. These fields are primarily meant for internal use by computer systems and represent nothing more than an alphabetic code. Therefore the allowable characters are restricted to a relatively small set that can be represented in 8-bit per character codes. This character set is selected from ISO/IEC 646. In addition to header fields, BCS characters can also be used for text data segments. Valid BCS character codes range from 20 through FF and line feed (0A), form feed (0B), and carriage return (0C).

4.1.3.3.2 Basic character set-numeric (BCS-N)

The range of allowable characters for BCS-N consists of the numbers '0' through '9' from the BMP block named 'BASIC LATIN', codes 30 through 39; plus sign, code 2B; minus sign, code 2D; decimal point, code 2E; and slash (/) , code 2F.

4.1.3.3.3 Basic character set-alphanumeric (BCS-A)

The range of allowable characters for BCS-A consists of the following: Space through Tilde, codes 20 through 7E (BMP block 'BASIC LATIN').

4.1.3.3.4 Universal multiple-octet coded character set (UCS)

The UCS is used for expressing text in many languages of the world as defined by ISO/IEC 10646. The specific character set selected from UCS shall be identified by profile. The profile shall identify the adopted form, the adopted implementation level and the adopted subset (list of collections and/or characters) in accordance with the structures defined in ISO/IEC 10646. When a profile defined UCS is used in a BIIF file, the coding shall contain an explicit declaration of identification of features (escape sequence) as specified in ISO/IEC 10646. When no declaration escape sequence is included, the default shall be that defined for BCS above. UCS-2 and UCS-4 are respectively, the two and four octet forms.

4.1.3.3.5 UCS Transformation Format 8 (UTF-8)

UTF-8 is the UCS Transformation Format 8 defined in ISO/IEC 10646-1, Amendment 2 that allows the use of multi-national character sets in BIIF. It is an alternative coded representation for all of the characters of the UCS. UCS characters from the BASIC Latin collection are represented in UTF-8 with single octet values ranging from 20 to 7E. Octet values 00 to 7F do not otherwise occur in the UTF-8 coded representation of any character. Therefore, the BASIC Latin collection is unchanged under a UTF-8 transformation. In UTF-8 each character has a coded representation that comprises a sequence of octets of length 1, 2, 3, 4, 5, or 6 octets. For all sequences of one octet, the most significant bit shall be a ZERO bit. All octets other than the first in a sequence are continuing octets. The number of ONE bits in the most significant positions of the first octet of the sequence determines how many continuing octets represent the international character. Selected header and subheader fields allow UTF-8 coded characters. Field lengths are fixed octet counts regardless of character encoding.

4.1.4 Logical structure of pixel storage

An image band is stored in row, column order, such that the column index is changing faster than the row index. The band index may be changing either faster than the column index or slower than the row index. The orientation of the pixels in the image array to up, down, left and right is defined by the metadata associated with the image. The image array structure is depicted in Figure 3. The origin of the image array I, pixel I(0,0), is at the upper left corner, and pixel I(R-1,C-1) is at the lower right corner. Within the r^{th} row, the pixels shall appear beginning on the left with I(r,0) and proceeding from left to right with I(r,1), I(r,2) and so on, ending with I(r,C-1).

4.1.4.1 Pixel coordinates

The black square in Figure 3 is a single pixel located by (r,c) where r = row number, and c = column number. Each pixel value for each band in an image can be uniquely identified by its (r, c) coordinate.

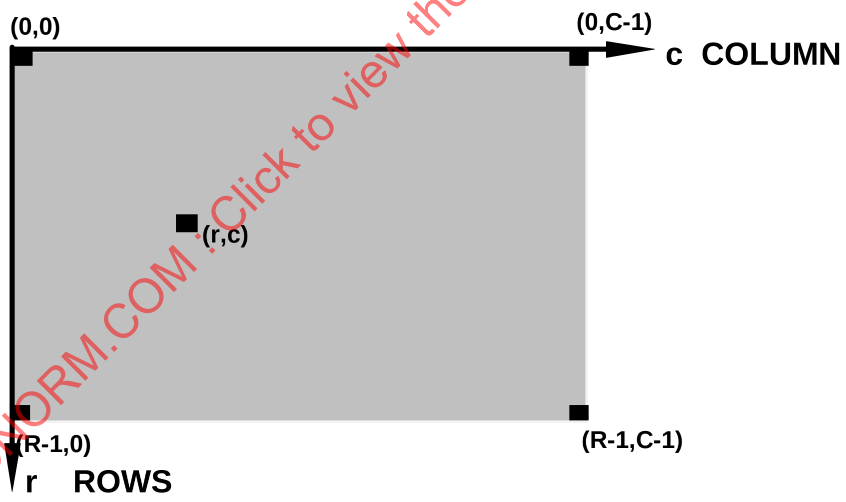


Figure 3 -- Image Array Structure

4.1.4.2 PIKS pixel coordinates

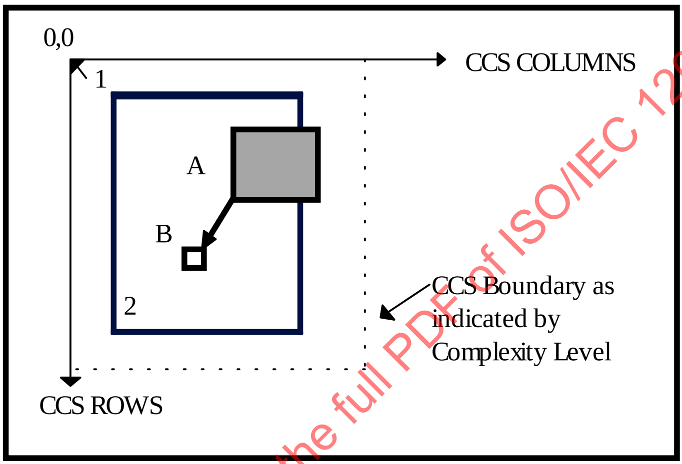
The Image Array pixel coordinate indexing (row, column) demonstrated in Figure 3 is typical for image scanning systems (e.g., line, sample), but is reversed from that used by PIKS. In PIKS, the first index is across columns. Implementers must take this into consideration when passing pixel coordinates through the PIKS Application Programming Interface (API). The pixel storage order is the same, only the coordinate indexing convention is reversed.

4.1.5 Common coordinate system

The BIIF Common Coordinate System (CCS) is the two dimensional coordinate space which shall be common for determining the placement and orientation of displayable data types (e.g. images, symbols, extension data, etc.) within a specific BIIF file and among correlated BIIF files which comprise an integrated product.

4.1.5.1 Common coordinate system structure

The Common Coordinate System structure can be conceived of as a virtual two dimensional drawing space with a coordinate system similar in structure to the lower right quadrant of the Cartesian coordinate system. The CCS has two perpendicular coordinate axes, the horizontal column axis and the vertical row axis as depicted in Figure 4. The positive directions of the axes are based on the predominate scan (column) and line (row) directions used by the digital imagery community. The intersection of the axes is designated as the origin point with the coordinates (0,0). Given the orientation of the axes in Figure 4, the positive direction for the column axis is from (0,0) and to the right; the positive direction for the row axis is from (0,0) downward. The quadrant represented by the positive column and positive row axes is the only coordinate space for which BIIF displayable data types may be located.



1. Common Coordinate System Origin (0,0)
2. Image
A. Inset Image
B. Inset Graphic (Box & Arrow)

Figure 4 -- Common coordinate system

4.1.5.2 Row and column coordinates

Images and symbols have designated points of references. For an image it is the first pixel (0,0). For a CGM symbol, it is the CGM virtual display coordinates (VDC) origin point. Symbols and images shall be placed in the CCS according to the CCS row and column coordinates placed in subheader location fields (e.g., ILOC, SLOC). The location coordinates of a specific image or symbol represent row and column offsets from either the CCS origin point when 'unattached', or the location point in the CCS of the data to which it is attached. Other means used to locate displayable data shall be directly correlated to row and column coordinates (e.g., displayable tagged record extension data might have geolocation data correlated with row and column indices). Note: When passing coordinate pairs to a PIKS API, the order of the coordinates need to be reversed by the application.

When location coordinates are relative to the CCS origin, they shall always have a positive value. When location coordinates are relative to the location coordinates of an item to which they are attached, both positive and negative offset values are possible. In all cases, the location coordinates selected for any data item shall ensure that none of the displayable item extends outside of the quadrant defined by the axes of the CCS.

4.1.5.3 CCS Boundaries

The upper and left boundaries of the CCS are explicitly constrained in the specification. The lower and right boundaries constraints are defined as one of the key attributes of the complexity level definition in a profile. Image and symbol segments shall be displayable and located such that no displayable portion of any image or subimage in a BIIF file extends beyond the CCS boundaries applicable to the file according to the selected complexity level.

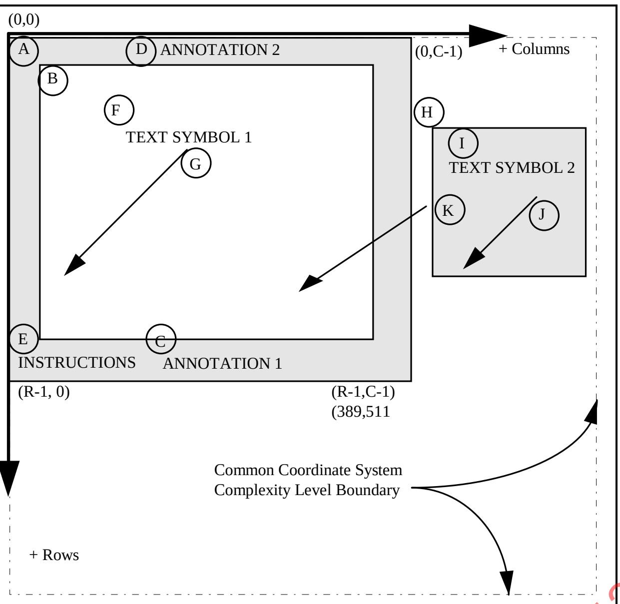
4.1.6 Display and attachment levels

Each image product shall be comprised of one or more data segments. The relative visibility of the various items in BIIF is recorded by use of the display level (the "DLVL" field in the standard data segment subheaders, specifically IDLVL for image, SDLVL for symbol). Groups of related items may be formed by use of the attachment level (the "ALVL" field in the standard data segment subheaders, specifically IALVL for image, SALVL for symbol).

4.1.6.1 Display levels

The order in which images and symbols are "stacked" shall be determined by their display level, not by their relative position within a BIIF file. Every image and symbol component in a BIIF file shall have a unique display level. This requirement allows "stacking" to be independent of data sequence or processing order.

Figure 5 illustrates a sample "output presentation" from a BIIF file that illustrates the effects of display level assignment. The display level of each segment shown on Figure 5 is indicated in the list of segments, where the list is in the order that the segments were placed in BIIF containing them. In the case shown, the segment with display level one is not an image but rather an opaque CGM rectangle (symbol data, not image data). Because the CGM rectangle is larger than the image (which, in this case, serves as the first overlay because its display level is two), it provides a border to the image. Following increasing DL value, the border is overlaid by the image being examined which, in turn, is overlaid by arrow one, which is in turn overlaid by the image inset, which is overlaid by the annotation, which is overlaid by the arrow annotation, etc. The AL values in Figure 4 refer to "Attachment Levels".



	Location Offsets			
	DL	AL	Row	Column
A. Border (Opaque Box)	001	000	00000	00000
B. Exploited Image	002	001	00025	00025
C. Annotation 1	999	000	00375	00156
D. Annotation 2	998	000	00030	00156
E. Instructions	997	000	00375	00008
F. Text Symbol 1	003	002	00080	00100
G. Arrow 1	004	003	00010	00065
H. Image Inset	005	002	00080	00400
I. Text Symbol 2	006	005	00025	00032
J. Arrow 2	007	006	00065	00070
K. Arrow 3	008	005	00070	00050

Figure 5 -- Display level and attachment level relationships

4.1.6.2 Attachment levels

The attachment level (AL) provides a way to associate items (images, symbols) so they may be treated together for certain operations such as moving, rotating, or displaying. The attachment level of an item shall be equal to the display level of the item to which it is "attached." Items can only be attached to existing items at a lower display level. This value is stored in the "ALVL" field (specifically IALVL for images, SALVL for symbols) of the item's subheader. The segment with the lowest display level (display level 001 in the example in Figure 5), must have an attachment level of zero. An attachment level of zero shall be interpreted as "unattached." Any other item may also have AL 000, that is, be unattached.

Figure 6 shows the attachment relationships of overlay items in Figure 5. When an overlay or base is edited (moved, deleted, rotated), all overlays attached to it, directly or indirectly, shall be affected by the same operation. For example, in Figure 6, if the inset being examined (DL 005, AL 002) was moved one centimeter to the left, the arrows (DL 008, AL 005, and DL 007, AL 006) and symbol (DL 006, AL 005) associated with the examined inset (DL 005, AL 002) would also be moved one centimeter to the left. Also note that because of the way the attachments have been constructed, if the symbol (DL 006, AL 005) were deleted, so would arrow 2 (DL 007, AL 006). However, if the image inset (DL 005, AL 002) were deleted, arrow (DL 004, AL 003) would not be deleted.

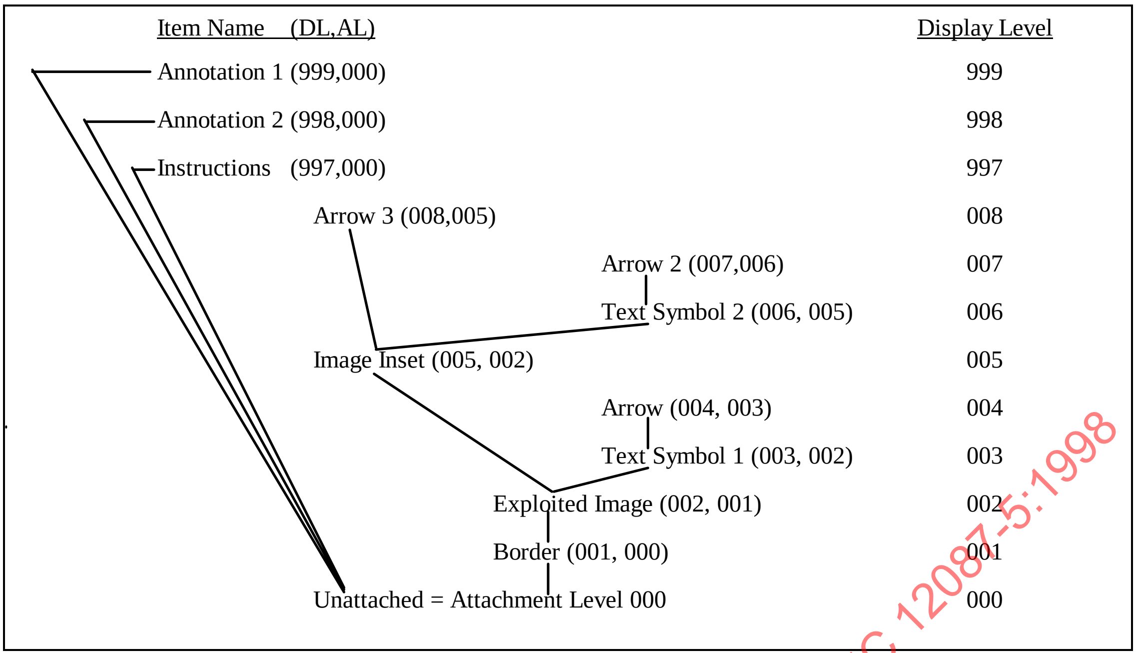


Figure 6 -- Display and attachment levels

4.2 Format

The format of the BIIF file consists of a header, followed by data type segments with their associated subheaders. The header specifies profile and structural information that allows proper interpretation of the rest of the header and subheaders. All headers and subheaders have their character data specified in the lexical constraints of BCS-A, BCS-N or UTF-8. Data in the text segments may be specified in other lexical levels of ISO 10646 as discussed in Subclause 4.1.3. The header is fixed in structure to support easy location and interpretation of the data. This is especially important for security-related information since this insures that security data is immediately accessible. Additionally, a fixed file structure allows an additional level of data integrity checking since octet counts are known in advance.

Each image, symbol, text, and extension data type included in BIIF shall be preceded by a "subheader" corresponding to that data item. This comprises a data segment. This subheader shall contain information pertaining to that particular data item and data type only. If no segments of a given type are included in BIIF, a subheader for that data type shall not be included in BIIF. A BIIF file may contain multiple segments of each data type (image, symbol, text, and extensions). All segments of a specific data type shall precede the segments of the next data type.

All data items and the associated subheader of a single type shall precede the first subheader for the next data type. Field values are to be read as a continuous stream as defined in the tables below.

4.2.1 Data recording formats.

The method of converting data into an octet string shall adhere to the following:

1) octet oriented data (BCS character string) shall be recorded with no change

2) the ordering sequence for bit oriented data (integers, reals) shall be recorded in the order where the left-most eight bits of remaining data are always output first. See Figure 7 where octet 0 is the most-significant (leftmost) octet within a multi-octet data structure.

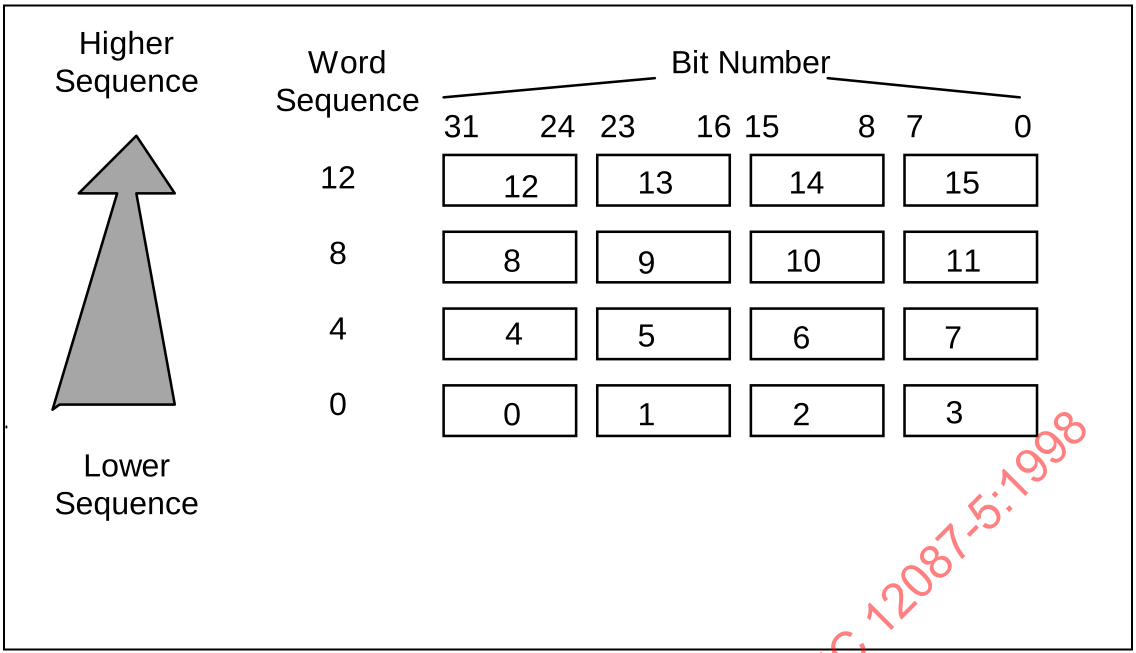


Figure 7 -- Octet sequence order

4.2.2 Encoding

The tables in the following subclauses specify the encoding structure of a BIIF file. BIIF specifies a single encoding consisting of both fixed and variable field sizes and a fixed field ordering. Compound elements (headers, subheaders, etc.) are constructed by packing primitive elements (fields, attributes) in a fixed order including or excluding certain condition dictates. The tables below provide the syntax/format for each of the primitive elements. A separate table is provided for each compound element. The following notation is used in the tabular entries of the five column tables which follow:

Column I. Field name: A short name used for references in the text descriptions.
Column II. Description: A short description of the meaning of the field, followed by a more detailed explanation as required.
Column III. Type A selection from the following codes:

R: Required element; structure (syntax) and meaning of this element shall be implemented precisely as specified in this standard. It must be present.

C: Conditional element; this element is omitted based on the value of its “dependent element”. For example, if NUMI (number of images) = 0, then no image components (image subheader, image data) are present in the file. A conditional field may or may not be present depending on the value of one or more preceding required fields.

PVV: Profile Variant-Value: A restriction on value range for this element is permitted in a profile. For example, a profile might be developed that supported only one image in the file. In this case, NUMI (number of images) would be constrained to the value one for all BIIF files conforming to this profile. The profile is required to identify constraints on the data value range.

PVU: Profile Variant - Unspecified: Both structure (syntax) and meaning are allowed to be defined by the profile; however, size and order within compound elements must be maintained in accordance with this standard. The PVU and PVV designation may be applied to either an R or C field and will be shown, for example, as R/PVV or C/PVU.

Column IV. CE/Size: Character Encoding: A = BCS-A; N =BCS-N, U8=UTF-8. Size: Equals the number of octets that are reserved for the field. Size is fixed and must be filled with valid data or the specified default. For those “Required” fields BCS-A “spaces” will be applied as default, for fields labeled BCS-N “leading zeros (0)” will be applied, for fields labeled U8, “spaces” will be applied (e.g., U8/80 - UTF8 encoded with a total length of 80 Octets, default is 80 “spaces”.)

Column V Value Range: Valid information must fall within the ranges identified and may be a range, an enumerated set, or a single value.

The specification of the fields in the various subheaders found within the BIIF file is provided in a series of tables. For each field in the subheader, the table includes a mnemonic field identifier, the field's name and description of the valid contents of the field, the field BCS type and size, and the range of values it may contain, and any constraints on the field's use. The Tagged Record Extension headers are defined in Table 8. The data that appears in all subheader information fields specified in the tables, including numbers, shall be represented using the basic character set with eight bits (one octet) per character. All field size specifications given for the header and subheader fields specify a number of octets.

4.2.3 Header

Each BIIF file shall begin with a header whose fields contain identification, origination information, security information, and the number and size of data items of each type contained in BIIF. Figure 8 depicts the header. It illustrates the types of information contained in the header and shows the header's organization as a sequence of groups of related fields. The expansion of the "Image Group" illustrates how the header's overall length and content may expand or contract depending on the number of data items of each type included in BIIF.

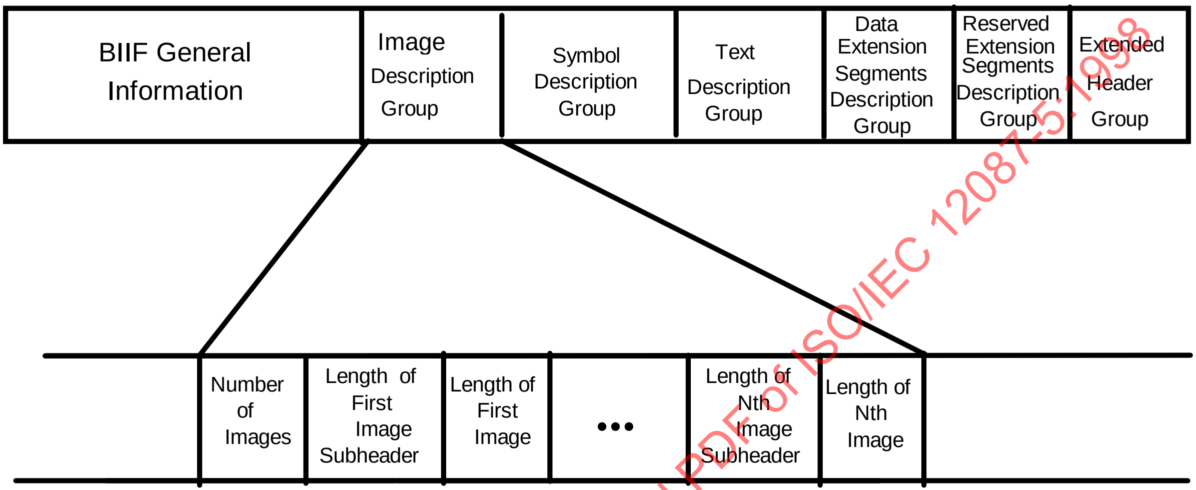


Figure 8 -- Header structure

Table 1 -- Header

FIELD	DESCRIPTION	TYPE	CE/SIZE	VALUE RANGE
FHDR	Profile Name A character string uniquely identified in the profile.	R/PVV	A/4	Only approved and registered profile short names.
FVER	Version A character string uniquely denoting the version.	R/PVV	A/5	Only approved and registered versions.
CLEVEL	Profile Complexity level A character string defined in the profile for BIIF complexity level (such as 00 through 99).	R/PVV	A/2	Only approved and registered profile complexity levels or 00 for non-hierarchical profiles.
STYPE	Standard Type Identifies the file as being formatted according to the BIIF standard.	R	A/4	“BF01”
OSTAID	Originator System ID This field shall contain an identification indicator of the originating system.	R/PVU	A/10	As specified in profile.
FDT	File Date & Time This field shall contain the date and time of the BIIF file origination. The time shall be specified in UTC (Z). A software implementation may display the time in any desired format; however, data recorded in a BIIF file shall be recorded in UTC.	R	N/14	CCYYMMDDhhmmss
FTITLE	File Title This field shall contain the title of the BIIF file. Any UTF-8 string is allowed.	R/PVU	U8/80	As specified in profile.
FSEC	File Security Profile Specific Parameters This field shall contain profile specific information for the image product security as defined in the profile.	R/PVU	U8/167	As specified in profile.
FSCOP	File Copy Number This field shall contain the copy number of the file.	R/PVV	N/5	00000 default, or actual number.
FSCPYS	File Number of Copies. This field shall contain the total number of copies of the file.	R/PVV	N/5	00000 default or actual count.
ENCRYP	Encryption Encryption codes and meaning as defined by profile.	R/PVU	A/1	0 = not encrypted, other codes as specified by profile.
OID	Originator's ID or other file information.	R/PVU	U8/45	As specified in profile.

Table 1 -- Header *(continued)*

FIELD	DESCRIPTION	TYPE	CE/SIZE	VALUE RANGE
FL	File Length This field shall contain the length in octets of the entire BIIF file including all headers, subheaders, and data.	R/PVV	N/12	"000000000388"- "999999999999"
HL	Header Length This field shall contain a valid length in octets of the BIIF header.	R	N/6	"000388"- "999999"
NUMI	Number of Images This field shall contain the number of separate images included in the BIIF file. This field shall be zero if and only if no images are included in the BIIF file.	R/PVV	N/3	"000"- "999"
LISH001	Length of 1st Image Subheader If the field NUMI contains a value of one or more, this field shall contain a valid length in octets for the subheader of the first image in the BIIF file. This field is conditional and shall be omitted if NUMI field contains zero.	C	N/6	"000439"- "999999"
LI001	Length of 1st Image This field shall contain a valid length in octets of the first image. This field is conditional and shall be omitted if NUMI field contains zero.	C	N/10	"0000000001"- "9999999999"
.....				
LISHn	Length of nth Image Subheader This field shall contain a valid length in octets for the nth image subheader, where n is the number of the image counting from the first image in order of the images' appearance in the BIIF file. This field shall occur as many times as specified in the NUMI field. This field is conditional and shall be omitted if NUMI field contains a value less than n. Possible values of n are 002 to 999.	C	N/6	"000439"- "999999"

Table 1 -- Header *(continued)*

FIELD	DESCRIPTION	TYPE	CE/SIZE	VALUE RANGE
LIn	Length of nth Image This field shall contain a valid length in octets of the nth image, where n is the image number of the image counting from the first image in order of the images' appearance in the BIIF file. If the image is compressed, the length after compression shall be used. This field shall occur as many times as specified in the NUMI field. This field is conditional and shall be omitted if NUMI field contains a value less then n. Possible values of n are 002 to 999.	C	N/10	"0000000001"- "9999999999"
NUMS	Number of Symbols This field shall contain the number of separate symbols included in the BIIF file. This field shall be zero if and only if no symbols are included in the BIIF file.	R/PVV	N/3	"000"-"999"
LSSH001	Length of 1st Symbol Subheader If the field NUMS contains a value of one or more, this field shall contain a valid length in octets for the subheader of the first symbol in the BIIF file. This field is conditional and shall be omitted if NUMS field contains zero.	C	N/4	"0258"-"9999"
LS001	Length of 1st Symbol This field shall contain a valid length in octets for the first symbol. This field is conditional and shall be omitted if NUMS field contains zero.	C	N/6	"000001"- "999999"
.....				
LSSHn	Length of nth Symbol Subheader This field shall contain a valid length in octets for the nth symbol subheader, where n is the number of the symbols counting from the first symbol in the order of the symbol's appearance in the BIIF file. This field shall occur as many times as specified in the NUMS field. This field is conditional and shall be omitted if NUMS contains zero. Possible values of n are 002 to 999.	C	N/4	"0258"-"9999"

Table 1 -- Header *(continued)*

FIELD	DESCRIPTION	TYPE	CE/SIZE	VALUE RANGE
LSn	Length of n^th^ Symbol This field shall contain a valid length in octets of the n^th^ symbol, where n is the symbol number of the symbol, counting from the first symbol in the order of the symbol's appearance in the BIIF file. This field shall occur as many times as specified in the NUMS field. This field is conditional and shall be omitted if NUMS field contains a value less then n. Possible values of n are 002 to 999.	C	N/6	"000001"- "999999"
NUMX	Reserved for Future Segment Types	R	N/3	"000"
NUMT	Number of Text Segments This field shall contain the number of separate text segments included in the BIIF file. The value is valid only if it is within the specified range. This field shall be zero if and only if no text items are included in the BIIF file.	R/PVV	N/3	"000"-"999"
LTSH001	Length of 1^st^ Text Subheader If the field NUMT contains a value of one or more, this field shall contain a valid length in octets for the subheader of the first text item in the BIIF file. This field is conditional and shall be omitted if NUMT field contains zero.	C	N/4	"0282"-"9999"
LT001	Length of 1^st^ Text Item This field shall contain a valid length in octets for the first text item. This field is conditional and shall be omitted if NUMT field contains zero.	C	N/5	"00001"-"99999"
.....				
LTSHn	Length of n^th^ Text Subheader This field shall contain a valid length in octets for the n^th^ text segment subheader, where n is the number of the text segments, counting from the first text segment in the order of the text segment's appearance in the BIIF file. This field shall occur as many times as specified in the NUMT field. This field is conditional and shall be omitted if NUMT field contains a value less then n. Possible values of n are 002 to 999.	C	N/4	"0282"-"9999"

Table 1 -- Header *(continued)*

FIELD	DESCRIPTION	TYPE	CE/SIZE	VALUE RANGE
LTn	Length of n^{th} Text item This field shall contain a valid length in octets of the n^{th} text item, where n is the number of the text segment, counting from the first text segment in the order of the text segment's appearance in the BIIF file. This field shall occur as many times as specified in the NUMT field. This field is conditional and shall be omitted if NUMT field contains less than n. Possible values of n are 002 to 999.	C	N/5	"00001"-"99999"
NUMDES	Number of Data Extension Segments This field shall contain the number of separate data extension segments included in the BIIF file. This field shall be zero if and only if no data extension segments are included in the BIIF file.	R/PVV	N/3	"000"-"999"
LDSH001	Length of 1^{st} Data Extension Segment Subheader If the field NUMDES contains a value of one or more, this field shall contain a valid length in octets for the subheader of the first data extension segment in the BIIF file. This field is conditional and shall be omitted if the NUMDES field contains zero.	C	N/4	"0200"-"9999"
LD001	Length of 1^{st} Data Extension Segment Data Field This field shall contain a valid length in octets for the data field of the first data extension segment. This field is conditional and shall be omitted if the NUMDES field contains zero.	C	N/9	"000000001"- "999999999"
.....				
LDSHn	Length of n^{th} Data Extension Segment Subheader This field shall contain a valid length in octets for the n^{th} extension segment subheader, where n is the number of the data extension segment counting from the first data extension segment in order of the data extension segment's appearance in the BIIF file. This field shall occur as many times as are specified in the NUMDES field. This field is conditional and shall be omitted if the NUMDES field contains a value less than n. Possible values of n are 002 to 999.	C	N/4	"0200"- "9999"

Table 1 -- Header *(continued)*

FIELD	DESCRIPTION	TYPE	CE/SIZE	VALUE RANGE
LDn	Length of nth Data Extension Segment Data Field This field shall contain a valid length in octets of the data field of the nth data extension segment, where n is the number of the data extension segment counting from the first data extension segment in order of the data extension segment's appearance in the BIIF file. This field shall occur as many times as are specified in the NUMDES field. This field is conditional and shall be omitted if the NUMDES field contains a value less than n. Possible values for n are 002 to 999.	C	N/9	"000000001"- "999999999"
NUMRES	Number of Reserved Extension Segments This field shall contain the number of separate reserved extension segments included in the BIIF file. This field shall be zero if and only if no reserved extension segments are included in the BIIF file.	R/PVV	N/3	"000"-"999"
LRESH001	Length of 1st Reserved Extension Segment Subheader If the field NUMRES contains a value of one or more, this field shall contain a valid length in octets for the subheader of the first reserved extension segment in the BIIF file. This field is conditional and shall be omitted if the NUMRES field contains zero.	C	N/4	"0200"-"9999"
LRE001	Length of 1st Reserved Extension Segment Data Field This field shall contain a valid length in octets for the data field of the first reserved extension segment. This field is conditional and shall be omitted if the NUMRES field contains zero.	C	N/7	"0000001"- "9999999"
.....				

Table 1 -- Header *(continued)*

FIELD	DESCRIPTION	TYPE	CE/SIZE	VALUE RANGE
LRESHn	Length of n^{th} Reserved Extension Segment Subheader This field shall contain a valid length in octets for the n^{th} reserved segment subheader, where n is the number of the reserved extension segment counting from the first reserved extension segment in order of the reserved extension segment's appearance in the BIIF file. This field shall occur as many times as are specified in the NUMRES field. This field is conditional and shall be omitted if the NUMRES field contains a value less than n. Possible values for n are 002 to 999.	C	N/4	"0200"-"9999"
LREn	Length of n^{th} Reserved Extension Segment Data Field This field shall contain a valid length in octets of the data field of the n^{th} reserved extension segment, where n is the number of the reserved extension segment counting from the first reserved extension segment in order of the reserved extension segment's appearance in the BIIF file. This field shall occur as many times as are specified in the NUMRES field. This field is conditional and shall be omitted if the NUMRES field contains a value less than n. Possible values for n are 002 to 999.	C	N/7	"0000001"- "9999999"
UDHDL	User Defined Header Data Length This field shall contain the length in octets of the entire UDHD field plus three (3) octets. The length is three (3) plus the sum of the lengths of all the Tagged Record Extensions (TREs) appearing in the UDHD field, since they are not separated from one another. A value of zero shall mean that no Tagged Record Extensions are included in the UDHD field. If a Tagged Record Extension is too long to fit in the UDHD field, it may be put in a Data Extension Segment (DES). See Section 4.2.8 on Tagged Record Extensions.	R/PVV	N/5	"00000" or "00003" - "99999"
UDHOFL	User Defined Header Overflow If present, this field shall contain "000" if the Tagged Record Extensions in UDHD do not overflow into a DES, or shall contain the sequence number of the DES into which they do overflow. This field shall be omitted if the field UDHDL contains zero.	C	N/3	"000-999"

Table 1 -- Header *(concluded)*

FIELD	DESCRIPTION	TYPE	CE/SIZE	VALUE RANGE
UDHD	User Defined Header Data If present, this field shall contain Tagged Record Extensions (TREs) as allowed by the profile. The length of this field shall be the length specified by the UDHDL field minus three (3) octets. The TREs shall appear one after the other with no intervening octets. This field shall be omitted if the field UDHDL contains zero.	C/PVU	TREs, Length specified by UDHDL minus 3	TREs as allowed by Profile
XHDL	Extended Header Data Length This field shall contain the length in octets of the entire XHD field plus three (3). The length is three (3) plus the sum of the lengths of all the Tagged Record Extensions appearing in the XHD field, since they are not separated from one another. A value of zero shall mean that no TREs are included in the XHD field. If a TRE is too long to fit in the XHD field, it may be put in a Data Extension Segment (DES).	R/PVV	N/5	"00000" or "00003"-"99999"
XHDLOFL	Extended Header Data Overflow If present, this field shall contain "000" if the Tagged Record Extensions in XHD do not overflow into a DES, or shall contain the sequence number of the DES into which they overflow. The XHDLOFL field shall be omitted if the XHDL field contains a value of zero.	C	N/3	"000-999"
XHD	Extended Header Data If present, this field shall contain Tagged Record Extensions as allowed by the profile. The length of this field shall be the length specified by the field XHDL minus three octets. Tagged record extensions shall appear one after the other with no intervening octets. This field shall be omitted if the field XHDL contains zero.	C/PVU	TREs, Length specified by XHDL minus three (3)	TREs as allowed by Profile

4.2.4 Image segment

4.2.4.1 Image subheader

In a BIIF file, the information describing an image segment is represented in a series of adjacent fields grouped into the image subheader followed by the image data. The field containing the image data is called the image data field. The image data field shall follow immediately the last field of the corresponding image subheader with no intervening special characters to designate the beginning of the image data field. Similarly, the image subheader of the first image shall follow immediately the last octet of data of the last field in the BIIF file header. The image subheader of successive images shall follow immediately the last octet of the image data field of the preceding image. Valid pixel data types, and their minimum levels of precision, are given in Table 2. The maximum level of precision is 96 bits as determined by the maximum value of the NBPP (Number of Bits Per Pixel Per Band) field in the image subheader. The NBPP field has a value in the range 0-96 bits unless constrained by profile. The table indicates the correspondence between BIIF pixel data types, PIKS pixel data types, and 12087-1 data types.

Table 2 -- Image pixel data type codes and descriptions

PVTYPE[1]	DESCRIPTION	Related PIKS Codes	12087-1 Data Types
B	1 bit Boolean values - On or off	BD	BC
INT	2 bit or greater unsigned integer	ND	NC
SI	2 bit or greater signed integer	SD	SC
R	Real arithmetic, 32-bit floating point representation	RD	RC
C	Complex arithmetic, values shall be represented with real and imaginary parts; each in 32-bit floating point representation and appearing in adjacent blocks, first real, then imaginary	CD	CC

[1]The 32 bit representation of real and complex numbers in the table is in accordance with IEEE 754 Standard for Binary Floating Point Arithmetic. Alternate representation for real and complex numbers can be identified through the registration process for inclusion in profiles.

4.2.4.2 Look-up tables (LUTS)

The BIIF provides a basic mechanism for associating simple look up tables in the image subheader for use in conjunction with the image data. The number of LUTS required depends on the Image Representation (IREP) value and the number of bands. Additional flexibility in defining and using look-up tables can be obtained through the use of PIKS object classes as defined in the TFS, Annex A. The following subclauses address common uses of look-up tables.

4.2.4.2.1 Gray scale look up tables (LUT)

The gray scale to be used in displaying each pixel of a gray scale image is determined using the image's LUT, if present. Multiple LUTS may be used to translate index values into multiple octet values. A LUT for a gray scale image shall comprise a one octet entry for each integer (the entry's index) in the range 0 to NELUT-1. The octets of the LUT shall appear in the file one after the other without separation. The entries shall occur in the index order, the first entry corresponding to index 0, the second to index 1, and so on, the last corresponding to index NELUT-1. The display shade for a pixel in the image shall be determined by using the image pixel value as an index into the LUT. The LUT value shall correspond to the display gray scale in a way specific to the display device. NELUT shall be equal to or greater than the maximum pixel value in the image to ensure that all image pixels are mapped to the display device.

4.2.4.2.2 Colour look-up tables (LUT)

Colour look-up tables consisting of 3 or 4 LUTS (depending on colour system used) allow the pixel values in an image band to be translated to a 3 or 4 band color system representation. Colour is represented according to the color system designated in the IREP field. For example, in the case of the RGB colour system representation, each LUT entry, if present, shall be composed of the output colour components red, green, and blue, appearing in the file in that order. There shall be a LUT entry for each pixel value in a particular band of a BIIF image (the entry's index of the LUT will range from 0 to 2^{NBPP}-1). The LUT entries shall appear in the file in increasing index order beginning with index 0. The display colour of an image pixel shall be determined by using the pixel value as an index into each LUT (red, green, blue). The corresponding values for red, green, blue shall determine the displayed colour in a manner specific to the display device. The colour component values may be any of the 256 pixel values associated with the band. Pseudo-colour (e.g., 8 bit per pixel colour images, IREP=RGB/LUT) contain three LUTs to correlate each pixel value with a designated true color value. Multi-spectral images may include colour LUTs in those bands which must be displayed independently of other bands.

Table 3 -- Image subheader

FIELD	DESCRIPTION	TYPE	CE/SIZE	VALUE RANGE
IM	BIIF Image Subheader This field shall contain the characters "IM" to identify the subheader as an image subheader.	R	A/ 2	"IM"
IID	Image ID This field shall contain an image identification.	R/PVU	A/10	As specified by profile.
IDATIM	Image Date & Time This field shall contain the date and time of the image origination.	R	N/14	CCYYMMDDhhmmss
IINFO	Image Information This field contains information about the main items of interest in the image.	R/PVU	U8/97	As specified by profile.
ISCSEC	Image Security Profile Specific Parameters This field shall contain profile specific information for the image segment security.	R/PVU	U8/167	As specified by profile.
ENCRYP	Encryption Encryption codes and meaning as defined by profile.	R/PVU	A/1	0 = not encrypted. Other codes as specified by profile.
ISORCE	Image Source This field contains the source of the image.	R/PVU	U8/42	As specified by profile.
NROWS	Number of Valid Rows in image This field contains the total number of rows of valid pixels in the image.	R/PVV	N/8	"00000001"-"99999999"
NCOLS	Number of Valid Columns in image This field contains the total number of columns of valid pixels in the image.	R/PVV	N/8	"00000001"-"99999999"

Table 3 -- Image subheader *(continued)*

FIELD	DESCRIPTION	TYPE	CE/SIZE	VALUE RANGE
PVTYPE	Pixel data representation type This field shall contain an indicator of the type of computer representation used for the value for each pixel for each band in the image.	R/PVV	A/3	As specified in profile selecting values from Table 2 or additional values as defined through the registration process.
IREP	Image Representation This field shall contain a valid indicator for the general kind of image represented by the data.	R/PVV	A/8	MONO, RGB, RGB/LUT, IHS, CMY, CMYK, YIQ, YUV, YCbCr, CIE, 1D, 2D, ND, MULTI, PIKS - additional values specified through the graphical item registration process.
ICAT	Image Category This field shall contain a valid indicator of the specific category (often revealing the nature of the collector or intended use).	R/PVU	A/8	The table footnote[1] lists existing categories - additional values added through the graphical item registration process.
ABPP	Actual bits-Per-Pixel Per Band This field shall contain the number of "valid bits" for the value in each band of each pixel without compression.	R/PVV	N/2	01-96 or as constrained by profile.
PJUST	Pixel Justification When ABPP is not equal to NBPP, this field indicates whether the most significant bits are left justified (L) or right justified (R).	R/PVV	A/1	L or R

[1]VIS - Visual imagery, SL - Side looking radar, TI - Thermal infrared, FL - Forward looking IR, RD - Radar, EO - Electro-optical, OP - Optical, HR - High resolution radar, HS - Hyperspectral, CP - Colour frame photo, BP - Black/white photo, SAR - Synthetic aperature radar, SARIQ - SAR radio hologram, IR - Infrared, MS - Multi-spectral, FP - Finger prints, MRI - Magnetic resonance imagery, XRAY - x-rays, CAT - CAT scan, MAP - Image map, PAT - Colour patch, LEG - Legend, DTEM - Elevation model data, MATR - general matrix data, LOCG - Location grids

Table 3 -- Image subheader *(continued)*

FIELD	DESCRIPTION	TYPE	CE/SIZE	VALUE RANGE
ICORDS	Image Coordinate Type This field shall contain a valid code indicating the coordinate system for the image.	R/PVV	A/1	"space" = None Other codes as defined in profile indicate the existence of the conditional field IGEOLO.
IGEOLO	Image Location utilized for representation of the coordinates of the image. This field is omitted if ICORDS = space.	C/PVU	A/60	As defined in profile.
NICOM	Number of Image Comments This field shall contain the valid number of 80 character blocks (ICOMn) that follow to be used as free text image comments.	R/PVV	N/1	0-9
ICOM1	Image Comment 1 This field, when present, shall contain free-form BCS-A text. This is the first comment field.	C/PVU	U8/80	
...				
ICOMn	Image Comment n This field, when present, shall contain free-form UTF-8 text. Continuation of the previous comment fields.	C/PVU	U8/80	
IC	Image Compression This field contains a valid profile defined code indicating the form of compression used in representing the image data. The character M is reserved for use to indicate that image data mask tables have been included in the image data field.	R/PVV	A/2	"NC"= Uncompressed "NM" = Uncompressed with mask table Additional codes defined in profile
COMRAT	Compression Rate Information This field shall be present and contain a code as defined by profile indicating the compression rate for the image. If the value in IC is C4 or M4, this field shall contain a value given in the form of n.nn representing the number of bits-per-pixel for the compressed image. Explanation of the compression rate for vector quantization can be found in Annex B. This field should be present when IC is a value other than NC or NM.	C/PVU	A/4	Field is omitted when "IC"="NC" or "NM". For C4/M4 the value is in the form n.nn representing the approximate number of bits per pixel for the compressed image. As defined in profile for other IC Codes.
NBANDS	Number of Bands This field shall contain the number of bands comprising the image. This field and the IREP field are interrelated and independent of the IMODE field.	R/PVV	A/1	"0"=Number of bands contained in the conditional XBANDS field. "1"-"9" Number of bands "T" Band definition is contained in TFS DES associated with the image segment in the overflow fields.

Table 3 -- Image subheader *(continued)*

FIELD	DESCRIPTION	TYPE	CE/SIZE	VALUE RANGE
XBANDS	Number of Multi-Spectral Bands When NBANDS contains the value 0, this field shall contain the number of bands comprising the multi-spectral images with greater than 9 bands. Otherwise this field is not present.	C/PVV	N/5	00010-99999.
IREPBAND1	1st Band Representation When NBANDS contains the value one, this field shall contain all spaces. This indicates that the image representation for the single band is that contained in the IREP field. In all other cases, this field shall contain a valid indicator of the interpretation of the first band correlated with the value of IREP as defined in the profile.	R/PVV	A/2	(Default is Spaces) All cases listed in the IREP field are represented. The table footnote[1] outlines the appropriate values. Additional values are specified through graphical item registration.
ISUBCAT1	1st Band Significance for Image Category The use of this field is profile-defined. Its purpose is to provide the significance of the first band of the image with regard to the specific category, ICAT, of the overall image. An example would be the wavelength of IR imagery.	R/PVU	A/6	Profile Defined. (Default is Spaces)
IFC1	1st Band Image Filter Condition Profile defined flag of first band filter.	R/PVU	A/1	Profile Defined. Default is "N" (None).
IMFLT1	1st Band Standard Image Filter Code Profile defined indicator of first band filter.	R/PVU	A/3	Profile Defined. Default is spaces.
NLUTS1	1st Band Number of LUTS This field shall contain the number of look-up tables associated with the 1st band of the image correlated with IREP.	R/PVV	N/1	"0" - "4"

[1] IREP	IREPBAND1	IREPBAND2	IREPBAND3	IREPBAND4	IREPBAND5	...
MONO	2 SPACES or M	N/A	N/A	N/A	N/A	...
RGB/LUT	2 SPACES or LU	N/A	N/A	N/A	N/A	...
RGB	R	G	B	N/A	N/A	...
IHS	I	H	S	N/A	N/A	...
CMY	C	M	Y	N/A	N/A	...
CMYK	C	M	Y	K	N/A	...
YIQ	Y	I	Q	N/A	N/A	...
YUV	Y	U	V	N/A	N/A	...
YCbCr	Y	Cb	Cr	N/A	N/A	...
CIE	X	Y	Z	N/A	N/A	...
1D	*	N/A	N/A	N/A	N/A	...
2D	*	*	N/A	NA	N/A	...
ND	*	*	*	*	*	...
MULTI	*	*	*	*	*	...
PIKS1-4	*	*	*	*	*	...

* Defined by Profile. In the MONO field, 2 spaces is equivalent to M. In the RGB/LUT field 2 spaces is equivalent to LU.
A profile can use 2 spaces in IREPBANDn to indicate a band not displayed and then use R, G, and B to identify the 3 bands which are to be displayed.
CIE is "Commission Internationale de l'Eclairage", CIE Publication #15, Colorimetry (1971 and 1986)

Table 3 -- Image subheader *(continued)*

FIELD	DESCRIPTION	TYPE	CE/SIZE	VALUE RANGE
NELUT1	1st Band Number of LUT Entries This field shall contain the number of entries in each of the look-up tables for the first band of data. This field shall be omitted if the value in NLUTS is zero.	C/PVV	N/5	"00001" - "65536"
LUTD11	Data of 1^{st} LUT for 1st Band This field shall be omitted if the first Band Number of LUTs is zero. Otherwise, this field shall contain the data defining the first look-up table for the first image band. This field supports only integer band data (PVTYPE=INT). Multiple LUTs may be used to translate the index value into multiple octet values.	C	Derived from Value NELUT1	Each entry in the look-up table is composed of one octet.
.....				
LUTD1m	Data of m^{th} LUT for 1st Band This field shall be omitted if the first Band Number of LUTs is zero. Otherwise, this field shall contain the data defining the m^{th} look-up table for the first image band.	C	Derived from Value NELUT1	Each entry in the look-up table is composed of one octet.
NOTE: The fields IREPBANDn through LUTDnm repeat the number of times indicated in the NBANDS field or the XBANDS field. Sequence continuation is indicated elsewhere by a series of dots (...).				
.....				
IREPBANDn	n^{th} Band Representation This field shall contain a valid indicator of the interpretation of the n^{th} band correlated with the value of IREP. Possible values for n are 00002 to 99999.	C/PVV	A/2	See IREPBAND1.
ISUBCATn	n^{th} Band Sub Category The use of this field is profile-defined. Its purpose is to provide the significance of the n^{th} band of the image with regard to the specific category, ICAT, of the overall image. An example would be the wavelength of IR imagery. Possible values for n are 00002 to 99999.	C/PVU	A/6	Profile Defined.
IFCn	n^{th} Band Image Filter Condition Profile defined flag of n^{th} band filter. Possible values for n are 00002 to 99999.	C/PVU	A/1	Profile Defined. Default is "N" (None).
IMFLTn	n^{th} Band Standard Image Filter Code Profile defined indicator of first band filter. Possible values for n are 00002 to 99999.	C/PVU	A/3	Profile Defined. Default is spaces.
NLUTSn	n^{th} Band Number of LUTS This field shall contain the number of look-up tables associated with the n^{th} band of the image correlated with IREP. Possible values for n are 00002 to 99999.	C	N/1	"0"-"4"

Table 3 -- Image subheader *(continued)*

FIELD	DESCRIPTION	TYPE	CE/SIZE	VALUE RANGE
NELUTn	n^{th} Band Number of LUT Entries This field shall contain the number of entries in each of the look-up tables for the n^{th} band of data. This field shall be omitted if the value in NLUTS is zero. Possible values for n are 00002 to 99999.	C	N/5	"00001"-"65536"
LUTDn1	This field shall be omitted if the n^{th} Band Number of LUTS is zero. Otherwise, this field shall contain the data defining the first look-up table for the nth image band. This field supports only INT band data (PVTYPE = INT). Possible values for n are 00002 to 99999 Multiple LUTs may be used to translate the index value into multiple octet values.	C/PVU	Derived from value of NELUTn	Binary LUT Values
.....				
LUTDnm	This field shall be omitted if the n^{th} Band Number of LUTs is zero. Otherwise, this field shall contain the data defining the m^{th} look-up table for the n^{th} image band. Each entry in the look-up table is composed of one octet, ordered from most significant bit to least significant bit representing a value from 0 to 255. This field supports only INT band data (PVTYPE=INT). Possible values for n are 00002 to 99999, m can have values 1-4.	C	Derived from value of NELUTn	Binary LUT Values
ISYNC	Image octet Alignment Code For uncompressed messages this field contains an indicator if end of row or column markers are used.	R/PVV	A/1	Default is "0" indicating no sync codes. Other values may be profile defined.
IMODE	Image Mode This field shall contain an indicator of whether the image bands are stored in the file sequentially or interleaved (by block, row or pixel).	R/PVV	A/1	B = Block Interleaved, P = Pixel Interleaved, S = Band Sequential, R = Row Interleaved.
NBPR	Number of Blocks Per Row This field shall contain the number of image blocks in a row of blocks (see 4.2.5.1) in the horizontal direction. If the image consists of only a single block, this field shall contain the value one.	R/PVV	N/4	"0001"-"9999"; or as constrained by Profile.
NBPC	Number of Blocks Per Column This field shall contain the number of image blocks in a column of blocks (see 4.2.5.1) in the vertical direction. If the image consists of only a single block, this field shall contain the value one.	R/PVV	N/4	"0001"-"9999", or as constrained by Profile.

Table 3 -- Image subheader *(continued)*

FIELD	DESCRIPTION	TYPE	CE/SIZE	VALUE RANGE
NPPBH	Number of Pixels Per Block Horizontal This field shall contain the number of pixels horizontally in each block of the image. It shall be the case that NBPR*NPPBV≥NCOLS.	R/PVV	N/4	"0001"-"9999", or as constrained by Profile.
NPPBV	Number of Pixels Per Block Vertical This field shall contain the number of pixels vertically in each block of the image. It shall be the case that NBPC*NPPB≥NROWS.	R/PVV	N/4	"0001"-"9999", or as constrained by Profile.
NBPP	Number of Bits Per Pixel Per Band This field shall contain the number of storage bits used for the value for each component of a pixel vector. The value in this field always shall be greater than or equal to Actual Bits Per Pixel (ABPP). For example, if 11-bit pixels are stored in 16 bits, this field shall contain 16 and Actual Bits Per Pixel shall contain 11.	R/PVV	N/2	"01"-"96", or as constrained by Profile.
IDLVL	Display level This field shall contain a valid value that indicates the graphic display level of the image relative to other associated BIIF segments in a composite display. The display level of each BIIF segment (image, or symbol) within the BIIF file shall be unique. The image, symbol, or segment in the BIIF file having the minimum display level shall have attachment level zero (unattached).	R	N/3	"001"-"999"
IALVL	Attachment Level This field shall contain a valid value that indicates the attachment level of the image. Valid values for this field are 000 and the display level value of any other image or symbol in the BIIF file. The image or symbol segment in the BIIF file having the minimum display level shall have attachment level zero.	R	N/3	"000"-"998"
ILOC	Image Location The image location is specified by specifying the location of the first pixel of the first line of the image. This field shall contain the image location offset from the ILOC or SLOC value of the segment to which the image is attached or from the origin of the CCS when the image is unattached (IALVL=000). A row or column value of 0 indicates no offset. Positive row and column values indicate offsets down and to the right, while negative row and column values indicate offsets up and to the left.	R	N/10	"RRRRRCCCCC" where for positive row and column values RRRRR= "00000"-"99999" and CCCCC= "00000"-"99999"; for negative row and column values RRRRR= "-0001" - "-9999" and CCCCC= "-0001"-"-9999"

Table 3 -- Image subheader *(continued)*

FIELD	DESCRIPTION	TYPE	CE/SIZE	VALUE RANGE
IMAG	Image Magnification Approximation This field shall contain the magnification (or reduction) factor of the image relative to the original source image. Decimal values are used to indicate magnification. Either decimal values or fractions may be used to indicate reduction. Fractions must be a slash (/) followed by an integer (with an implied preceding one). For example: “2.3” and “2.30” both indicate that the original image has been magnified by a factor of 2.3; while “0.5”, “.5”, “.50”, “.500”, and “/2” all indicate that the image has been reduced by half.	R	A/4	.001-9999 /1-/999 Value shall be a fraction (a slash followed by an integer) or a decimal value. Field shall be filled to the right with spaces. The default value is “1.0 ”
UDIDL	User Defined Image Data Length This field shall contain the length in octets of the entire UDID field plus three (3) octets. The length is three (3) plus the sum of the lengths of all the Tagged Record Extensions (TREs) appearing in the UDID field, since they are not separated from one another. A value of zero shall mean that no Tagged Record Extensions are included in the UDID field. If a Tagged Record Extension is too long to fit in the UDID field, it may be put in a Data Extension Segment (DES). See Section 4.2.8 on Tagged Record Extensions.	R/PVV	N/5	“00000” or "00003"-"99999"
UDOFL	User Defined Overflow If present, this field shall contain “000” if the Tagged Record Extensions in UDID do not overflow into a DES, or shall contain the sequence number of the DES into which they overflow. This field shall be omitted if the field UDIDL contains zero.	C	N/3	”000”-”999”
UDID	User Defined Image Tagged Data If present, this field shall contain Tagged Record Extensions as allowed by profile. The length of this field shall be the length specified by the field UDIDL minus three (3) octets. Tagged record extensions in this field for an image shall contain information pertaining specifically to the image. The TREs shall appear one after the other with no intervening octets. This field shall be omitted if the field UDIDL contains zero.	C/PVU	Tagged Record Extension - Length is specified by UDIDL minus three (3)	TREs as allowed by profile.

Table 3 -- Image subheader *(concluded)*

FIELD	DESCRIPTION	TYPE	CE/SIZE	VALUE RANGE
IXSHDL	Extended Subheader Data Length This field shall contain the length in octets of the entire field IXSHD plus three (3) octets. The length is three (3) plus the sum of the length of all Tagged Record Extensions appearing in the IXSHD field, since they are not separated from one another. A value of zero shall mean that no TREs are included in the IXHD field. If a TRE is too long to fit in the IXSHDL field, it shall be put in a Data Extension Segment (DES).	R/PVV	N/5	"00000" or "00003"-"99999"
IXSOFL	Extended Subheader Overflow If present, this field shall contain "000" if the Tagged Record Extensions in IXSHD do not overflow into a DES, or shall contain the sequence number of the DES into which they do overflow. The IXSOFL field shall be omitted if the field IXSHDL contains value of zero.	C	N/3	"000"-"999"
IXSHD	Extended Subheader Data If present, this field shall contain Tagged Record Extensions as allowed by the profile. The length of this field shall be the length specified by the field IXSHDL minus three octets. Tagged record extensions shall appear one after the other with no intervening octets. This field shall be omitted if the field XHDL contains zero.	C/PVU	Tagged Record Extensions length specified by IXSHDL minus three(3)	TREs as allowed by profile.

4.2.5 Image data field format

The following clauses address the structure of the image data within the image data field of the image segment except for the case when the IREP field has the value "PIKS". When IREP is "PIKS" , the TFS (described in Annex A) will include a PIKS OBJECT CLASS "PIKS_IMAGE" which defines the format and structure of the image data.

4.2.5.1 Blocked images

Image blocking extends the image model for BIIF presented above to support the representation of an image in terms of an orderly set of subimages (or subarrays) called blocks. A blocked image is analogous to a rectangular tiled floor; regard the overall floor as the image and each individual tile as a block. The performance of an imagery implementation can be potentially improved by "blocking" large images; that is, ordering the pixel values in the file as a series of concatenated pixel arrays (see Figure 9).

4.2.5.1.1 Block size

The number of rows and columns of pixels in each block must be less than or equal to the number of rows and columns of pixels in the overall image. Each block within an image array shall be the same size.

4.2.5.1.2 Block ordering

For recording purposes the image blocks are ordered sequentially by row (of blocks). In Figure 9, the recording order would be B0,0; B0,1; B0,2; B0,3; B1,0; B1,1; B1,2; B1,3; B2,0; B2,1; B2,2; B2,3. The first pixel of each succeeding block immediately follows the last pixel of the preceding block. Although the pixel values are placed in the file as a series of arrays (blocks), the coordinate used to reference any specific pixel remains the same as if the image were not blocked. For example, if each of the blocks in Figure 9 have NPPBV=NPPH=1024, the second pixel in B(0,1) has the coordinate (0,1025) vice the internal index (0,1) of the subarray.

4.2.5.1.3 Block divisibility condition

If the size of the overall image is not an integer multiple of the number of rows or columns in the image, the image shall be padded to an appropriate number of rows and columns so the divisibility condition is met by adding rows or columns to the bottom and right of the image (see Figure 9a and Figure 9b).

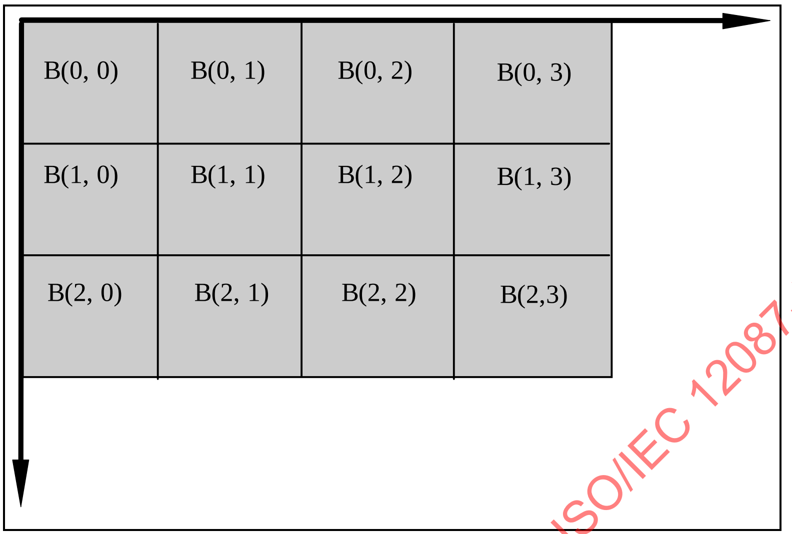


Figure 9a -- A blocked image

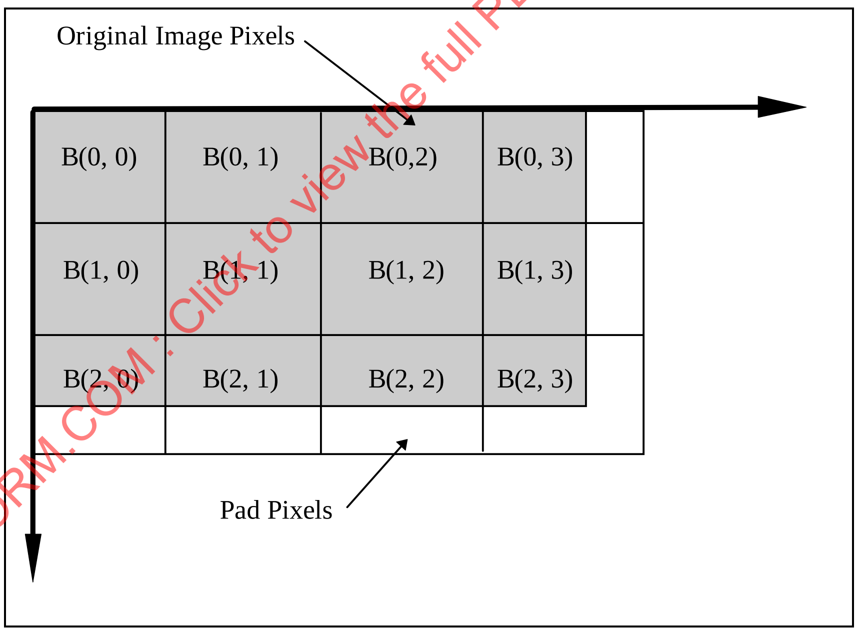


Figure 9b -- A blocked padded image

4.2.5.2 Image data masking

BIIF provides an option to include a blocked image mask table and a pad pixel mask table as part of the image data field immediately following the last octet of the image subheader. The presence of these tables is signaled when the value in the IC field contains the character "M".

4.2.5.2.1 Blocked image masking

In some instances, a blocked image may have a considerable number of empty blocks. This might occur when a rectangular image is not north aligned, but has been rotated to a north up orientation (see Figure 10). In this case, it is sometimes useful to not record or transmit empty blocks. However, if empty blocks are not recorded/transmitted, the image loses its logical structure as an image with n x m blocks. In order to preclude the loss of logical structure and to allow the exclusion of empty blocks, an image data mask table structure has been defined. The image data mask table is defined in Table 4. The field in the mask corresponding to a block tells whether or not the block is recorded and identifies the location of the block data if it is recorded. This allows an application to correctly reconstruct the image. In Figure 10, the recording order would be B0,0; B0,1; B0,2; B1,0; B1,1; B1,2; B1,3; B2,0; B2,1; B2,2; B2,3; B3,1; B3,2; B3,3. The blocked image mask would identify the locations of the recorded image blocks. If the image is band sequential (IMODE="S"), multiple block image masks shall be arranged in the same order as the image bands, with each mask containing NBPR x NBPC records. Block image masks can be used in conjunction with a pad pixel mask, as described below. A block image mask may also be used to provide random access within the blocked image data for large images even if all blocks are recorded.

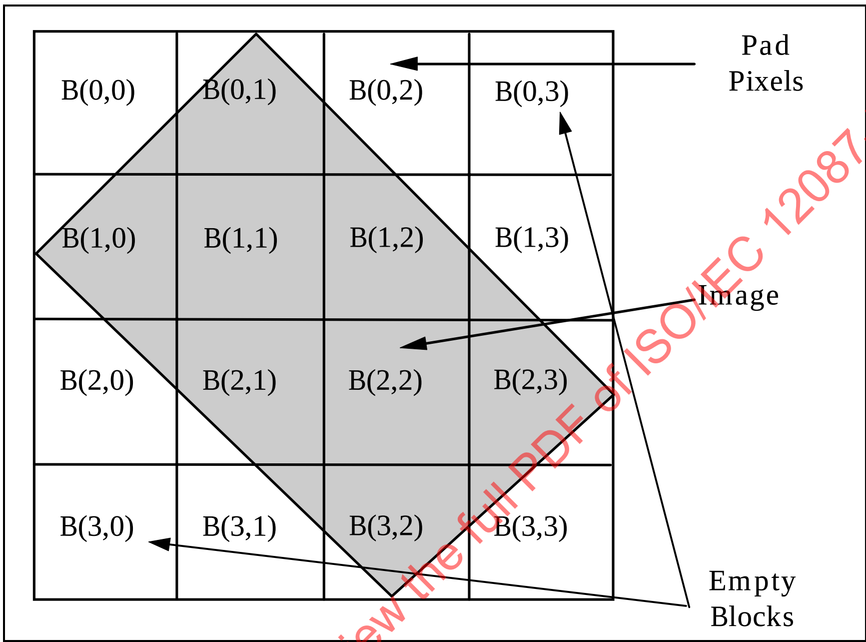


Figure 10 -- A blocked padded image with empty blocks

4.2.5.2.2 Pad pixel masking

In addition to empty image blocks, Figure 10 also demonstrates that a significant number of pad pixels may be needed to "fill" an image to the nearest block boundary. In the example in Figure 10, the locations of image blocks B0,0, B0,1; B0,2; B1,0; B1,2, B1,3, B2,0, B2,1, B2,3, B3,1, B3,2, and B3,3 would be recorded, indicating that these blocks have pad pixels. B0,3; B1,1; B2,2; and B3,0 do not have pad pixels because B0,3 and B3,0 are empty and B1,1 and B2,2 are full image blocks. If the image is band sequential (IMODE="S"), the pixel masks shall be arranged in the same order as the image bands, with each mask containing NBPR x NBPC records. The output pixel code which represents pad pixels is identified within the Image Data Mask table by the Pad Output Pixel Code field (TPXCD). The length in bits of this code is identified in the Output Pixel Code Length field (TPXCDLNTH). Although this length is given in bits, the actual TPXCD value is stored in an integral number of octets. When the number of bits used by the code is less than the number available in the TPXCD field (for example, a 12 bit code stored in two octets), then the code will be justified in accordance with the PJUST field in the Image Subheader. When an application identifies pad pixels, it may replace them with a user defined value (for example, a light blue background) at the time of presentation except when the value of TPXCD is zero (0). When the TPXCD value is zero, the pad pixel will be treated as "transparent" for presentation. The application may also choose to ignore pad pixels in histogram generation. In any case, pad pixels are not valid data, and should not be used for interpretation. Consequently, the value used for pad pixels in a masked image shall not appear within the bounds of significant pixel values of the image.

4.2.5.2.3 Image data mask table

The image data mask table is a conditional data structure included in the image data stream for masked images (IC value contains an M). The image data mask table is not recorded for non-masked images (IC value without an M). When an image mask table is used, the first octet of the image data is offset from the beginning of the image data area by the length of the image data mask table; and empty image blocks are not recorded/transmitted in the image data area. If the image is band sequential (IMODE="S"), the multiple block image and/or pixel masks shall be arranged in the same order as the image bands, with each mask containing NBPR x NBPC records. All block image masks will be recorded first, followed by all pad pixel masks. Since the image data mask tables are in the image area, the data recorded/transmitted there are binary. The structure of the image data mask table is defined in detail in Table 4.

Table 4 -- Image data mask table

FIELD	NAME	TYPE	SIZE-Octets	VALUE RANGE
IMDATOFF	Blocked Image Data Offset - This field is included if the IC value contains the character M. It identifies the offset from the beginning of the Image Data Mask Table to the first octet of the blocked image data. This offset, when used in combination with the offsets provided in the BMR fields, can provide random access to any recorded image block in any image band.	C	4	Unsigned Integer: 0 to 2^{32} - 1 (0x00000000-0xFFFFFFFF)
BMRLNTH	Block Mask Record Length - This field is included if the IC value contains the character M. It identifies the length of each Block Mask Record in octets. The total length of the Block Mask Records is equal to BMRLNTH x NBPR x NBPC x NBANDS. If all of the image blocks are recorded, this value may be set to 0, and the conditional BMR fields are not recorded/ transmitted. If this field is present, but coded as 0, then a pad pixel mask is included.	C	2	Unsigned Integer; 0=No Block mask; 4=Block mask present with each record length is 4 octets (0=0x0000) (4=0x0004)
TMRLNTH	Pad Pixel Mask Record Length - This field is included if the IC value contains the character M. It identifies the length of each Pad Pixel Mask Record in octets. The total length of the Pad Pixel Mask Records is equal to TMRLNTH x NBPR x NBPC x NBANDS. If none of the image blocks contain pad pixels, this value is set to 0, and the conditional TMR fields are not recorded/ transmitted. For IC value of M3, the value is set to 0. If this field is present, but coded as 0, then a Block Mask is included.	C	2	Unsigned Integer; 0=No pad pixel mask; 4=Pad pixel mask present , each record length is 4 octets (0=0x0000) (4=0x0004)
TPXCDLNTH	Pad Output Pixel Code Length This field is included if the IC value contains the character M. It identifies the length in bits of the Pad Output Pixel Code. If coded as 0, then no pad pixels are present, and the TPXCD field is not recorded. The length of the TPXCD field is next highest number of octets which can contain the number of bits identified in the TPXCDLNTH field. For Example, a TPXCDLNTH value of 12 would be stored in a TPXCD field of two octets.	C	2	Unsigned Integer; 0=No pad pixels; or Pad pixel code length in bits

Table 4 -- Image data mask table *(continued)*

FIELD	Name	TYPE	SIZE-Octets	VALUE RANGE
TPXCD	Pad Output Pixel Code - This field is included if the IC value contains the character M, and TPXCDLNTH is not 0. It contains the output pixel code that represents a pad pixel in the image. This value is unique within the image, and allows the user to identify pad pixels. The pad pixel output code length is determined by TPXCDLNTH, but the value is stored in a maximum of two octets. If the number of bits used by TPXCD is less than the number of bits available for storage, the value shall be justified in accordance with the PJUST field in the image subheader.	C	The length of the TPXCD field is the next highest number of octets which can contain the number of bits identified in the TPXCDLNTH field. (1 or 2 Octets)	Unsigned Integer; 0 to 2^n - 1 where n = TPXCDLNTH
BMR0BND1	Block Mask Record 0. Band 1 This field shall contain the first Block Mask Record of band 1. It is recorded/transmitted only if the BMRLNTH field is not 0. The field shall contain an offset in octets from the beginning of the Blocked Image Data to the first octet of block 0 of band 1 (this value should be 0) if block 0 is recorded/transmitted, or 0xFFFFFFFF if block 0 of band 1 is not recorded/transmitted in the image data.	C	4	Unsigned Integer; Offset in octets from the beginning of Blocked Image Data to the first octet of block 0 of band 1 (usually 0); 0xFFFFFFFF if the block is not recorded.
....				
BMRnBND1	Block Mask Record n, Band 1 This field shall contain the n^{th} Block Mask Record of band 1. It is recorded/transmitted only if the BMRLNTH field is not 0. The field shall contain an offset in octets from the beginning of the blocked Image Data to the first octet of block n of band 1 if block n is recorded or transmitted, or 0xFFFFFFFF if block n of band 1 is not recorded or transmitted in the image data. The number of BMR records for this band is NBPR x NBPC.	C	4	Unsigned Integer; Offset in octets from the beginning of Blocked Image Data to the first octet of block nn of band 1; 0xFFFFFFFF if the block is not recorded.
....				

Table 4 -- Image data mask table *(continued)*

FIELD	Name	TYPE	SIZE-Octets	VALUE RANGE
BMR0BNDm	Block Mask Record 0, Band m This field shall contain the nth Block Mask Record of band m. It is recorded/transmitted only if the BMRLNTH field is not 0. The field shall contain an offset in octets from the beginning of the Blocked Image Data to the first octet of block 0 of band m if block 0 of band m is recorded or transmitted, or 0xFFFFFFFF if block 0 of band m is not recorded or transmitted in the image data. The number of BMR records for this band is NBPR x NBPC. Possible values for m are 00002 to 99999.	C	4	Unsigned Integer; Offset in octets from the beginning of Blocked Image Data to the first octet of block 0 of band m; 0xFFFFFFFF if the block is not recorded.
....				
BMRnBNDm	Block Mask Record n, Band m This field shall contain the nnth Block Mask Record of band m. It is recorded/transmitted only if the BMRLNTH field is not 0. The field shall contain an offset in octets from the beginning of the Blocked Image Data to the first octet of block n of band m if block n of band m is not recorded or transmitted in the image data. The number of BMR records for this band is NBPR x NBPC. Possible values for m are 00002 to 99999.	C	4	Unsigned Integer; Offset in octets from the beginning of Blocked Image Data to the first octet of block n of band m; 0xFFFFFFFF if the block is not recorded.
TMR0BND1	Pad Pixel Mask Record 0, Band 1 This field shall contain the first Pad Pixel Mask Record for band 1. It is recorded/transmitted only if the TMRLNTH field is not 0. The field shall contain an offset in octets from the beginning of the blocked Image Data to the first octet of block 0 of band 1 if block contains pad pixels, or 0xFFFFFFFF to indicate that this block does not contain pad pixels.	C	4	Unsigned Integer; Offset in octets from the beginning of Blocked Image Data to the first octet of block 0 of band 1; 0xFFFFFFFF if the block does not contain pad pixels.
....				
TMRnBND1	Pad Pixel Mask Record n, Band 1 This field shall contain the nth Pad Pixel Mask Record for band 1. It is recorded/transmitted only if the TMRLNTH field is not 0. The field shall contain an offset in octets from the beginning of the Blocked Image Data to the first octet of block n of band 1 if block n contains pad pixels, or 0xFFFFFFFF to indicate that this block does not contain pad pixels. The number of TMR records for band 1 is NBPR x NBPC.	C	4	Unsigned Integer; Offset in octets from the beginning of Blocked Image Data to the first octet of block n of band 1; 0xFFFFFFFF if the block does not contain pad pixels.
....				

Table 4 -- Image data mask table *(concluded)*

FIELD	Name	TYPE	SIZE Octets	VALUE RANGE
TMR0BNDm	Pad Pixel Mask Record 0, Band m This field shall contain the first Pad Pixel Mask Record for band m. It is recorded/transmitted only if the TMRLNTH field is not 0. The field shall contain an offset in octets from the beginning of the Blocked Image Data to the first octet of block 0 of band m if block 0 contains pad pixels, or 0xFFFFFFFF to indicate that this block does not contain pad pixels. Possible values for m are 00002 to 99999.	C	4	Unsigned Integer; Offset in octets from the beginning of Blocked Image Data to the first octet of block 0 of band m; 0xFFFFFFFF if the block does not contain pad pixels.
....				
TMRnBNDm	Pad Pixel Mask Record n, Band m This field shall contain the nth Pad Pixel Mask Record for band m. It is recorded or transmitted only if the TMRLNTH field is not 0. The field shall contain an offset in octets from the beginning of the Blocked Image Data to the first octet of block n of band m if block n contains pad pixels, or 0xFFFFFFFF to indicate that this block does not contain pad pixels. The number of TMR records for band m is NBPR x NBPC. Possible values for m are 00002 to 99999.	C	4	Unsigned Integer; Offset in octets from the beginning of Blocked Image Data to the first octet of block n of band m; 0xFFFFFFFF if the block does not contain pad pixels.

4.2.5.3 Compressed image data format

The format of the image data placed in the image data field after compression is provided with the description of BIIF image compression algorithms in the appropriate Standards Documents selected by the profile (see Clause 2). The data resulting from image compression is located in the image data field of the image segment.

4.2.5.4 Uncompressed image data format

The order in which pixel values of a single band image are stored is fixed. When an image has more than one band, several options are available for the order in which pixel values are stored. The option used is indicated by the IMODE field in the image subheader. The following subclauses describe the possibilities within this format. In describing the encoding of image data, the BIIF display convention is invoked freely for ease of expression. Let the image to be encoded be denoted by I, and assume I has R rows and C columns. Let I have n bands; that is, each pixel is an n-vector, the i^{th} value of which is the value for that pixel location of the i^{th} band of the image. Let N denote the number of bits-per-pixel-per-band. Thus, there are n*N bits-per-pixel. Let I be blocked with H blocks per row and V blocks per column. Note that special cases such as single band images and single block images are included in this general image by setting n=1, and H=V=1, respectively.

4.2.5.4.1 Single band image uncompressed data format

For single band images, where n=1, there is only one order for storing pixels. The field IMODE in the image subheader shall be set to B for this case. The blocks (one or more) shall be stored, one after the other, starting with the upper left block and proceeding first left to right across rows of blocks, one row of blocks after the other, top to bottom. Image data within each block shall be encoded as one continuous bit stream, one pixel value after another, beginning with the N bits of the upper left corner pixel, I(0,0), followed by the N bits of I(0,1) and so on until all pixels from the first row in the block are encoded. These shall be followed immediately by the N bits of data for pixel I(1,0) continuing from left to right along each row, one row after another from the top of the block to the bottom. The last octet of each block's data is zero-filled (if necessary) to the next octet boundary, but all other octet boundaries within the block are ignored. See Table 2 for the specification of the bit representation of pixel values.

4.2.5.4.2 Multiple band image uncompressed data format

For multiple band images, there are four orders for storing pixels.

4.2.5.4.2.1 Band sequential

The first case is "band sequential", in which each band is stored contiguously, starting with the first band, one after the other, until the last band is stored. Within each band the data shall be encoded as if it were a single band image with multiple blocks. The field IMODE in the image subheader shall be set to S for this case. The IMODE S is only valid for images with multiple blocks and multiple bands. (For single block images, this case collapses to the “band interleave by block” case, where IMODE is set to B.)

4.2.5.4.2.2 Band interleaved by pixel

The ordering mechanism for this case stores the pixels in a block sequential order in which each block is stored contiguously, starting with the upper left block and proceeding first left to right across rows of blocks, one row of blocks after the other, top to bottom. Within each block, the n*N bits of the entire pixel vector are stored pixel-by-pixel in the same left to right, top to bottom pixel order as described in 4.1.5.5.2.1. The n*N bits for a single pixel are stored successively in this order: the N bits of the first band followed by the N bits of the second band and, so forth, ending with the N bits of the last band. Each block shall be zero-filled to the next octet boundary when necessary. The field IMODE in the image subheader shall be set to P for this storage option. See Table 2 for the specification of the bit representation of pixel values for each band.

4.2.5.4.2.3 Band interleaved by block

The ordering mechanism for this case stores the pixels in a block sequential order where each block is stored contiguously, starting with upper left block and proceeding first left to right across rows of blocks, one row of blocks after the other, top to bottom. Within each block, the data from each band is stored starting with the first band, one after the other until the last band is stored. Each block shall be zero-filled to the next octet boundary when necessary. The field IMODE in the image subheader shall be set to B for this storage option. See Table 2 for the specification of the bit representation of pixel values for each band.

4.2.5.4.2.4 Band interleaved by row

The ordering mechanism for this case stores the pixel values of each band in row sequential order. Within each block, all pixel values of the first row of the first band are followed by pixel values of the first row of the second band continuing until all values of the first row are stored. The remaining rows are stored in a similar fashion until the last row of values has been stored. Each block shall be zero filled to the next octet boundary when necessary. The field IMODE shall be set to R for this option.

4.2.5.5 Vector quantized data

Vector quantization is a structuring mechanism for use on multiband, colour, and gray scale scanned maps and imagery. The fundamental concept of vector quantization is to represent the image using a clustering technique to develop a codebook of quantized values. The indices of the codebook replace the image data in the BIIF file. The decoding mechanism is fully discussed in Annex B of this document.

4.2.5.6 Number of bands (NBANDS)

The NBANDS field in the image subheader shall define how many bands of imagery data are included in the BIIF file. If the value of this field is 0-9, the bands are homogeneous and are processed identically in accordance with parameters defined in the image subheader. For images greater than 9 bands, the NBANDS field is set to zero(0) and the number of bands is represented in the conditional field, XBANDS. When bands are not homogeneous (e.g., mix of spectral and time values), the value of NBANDS is set to T and the image band parameters are expanded by the Data Extension Segment described in the Transportable File Structure, Annex A, for use of PIKS object classes (e.g., PIKS_IMAGE).

4.2.5.7 PIKS Objects

BIIF provides for the transport of PIKS image objects and PIKS image-related non-image objects for applications in which the BIIF basic image and non-image data objects do not provide desired functionality. The transport mechanism for PIKS objects shall be the Transportable File Structure. The PIKS data objects are semantically specified in ISO/IEC 12087-1, Clause 5.4.

Table A.18 lists the size limits for PIKS parameters (BP, NP, SP, RP, CP, and CS) and pixel (BD, ND, SD, RD, CD) data types. In some cases, the minimum size limit for a PIKS pixel data type is greater than the minimum size limit for the corresponding BIIF pixel data type. If a BIIF image is imported into a PIKS application, it is possible that pixels may have to be padded to

PIKS minimum lengths. Table A.18 lists the data structures of the PIKS image and non-image data objects derived from ISO/IEC 12087-1, Clause 5.4. These objects include:

PIKS_IMAGE
PIKS_HISTOGRAM
PIKS_LOOKUP_TABLE
PIKS_MATRIX
PIKS_NEIGHBOURHOOD
PIKS_PIXEL_RECORD
PIKS_STATIC_ARRAY
PIKS_TUPLE
PIKS_ROI_COORDINATE
PIKS_ROI_ELLIPTICAL
PIKS_ROI_POLYGON
PIKS_ROI_RECTANGLE
PIKS_VALUE_BOUNDS

PIKS images provide expanded capability for representation from the "Baseline" capabilities of BIIF (e.g., baseline BIIF has homogeneous pixel values for monochrome or colour images while PIKS images may have heterogeneous values of up to five dimensions). The PIKS dimensions are in the following slice order:

x-y Spatial slice
z Volumetric slices
t Temporal slices
b Colour or multispectral slices

The PIKS image object data shall be placed in the transport data stream immediately following the image subheader, as specified in Table 3. The PIKS non-image object data shall be placed in the transport data stream immediately following the data object attributes for each object, as specified in Table A.19.

4.2.5.8 Image representation

The image representation (IREP) field contains a valid indicator for the general kind of image represented by the data. Valid representation indicators are MONO for monochrome; RGB for red, green, or blue true colour, RGB/LUT for mapped colour; 1D for monoband matrix/grid data; 2D for two dimensional data in support of location grids; ND for multiband matrix/grid data; and MULTI for multiband imagery. In addition, compressed imagery can have this field set to YCbCr when compressed in the ITU-R Recommendation BT.601-5 colour space using JPEG (field IC="C3"). This field should be used in conjunction with ICAT, ISUBCATn, and IREPBANDn fields to interpret the significance of each band in the image. Table 5 shows representative examples of IREP and some of its associated fields.

Table 5 -- Representative examples of IREP and associated fields

IREP	PVTYPE	NBPP	ABPP	NBANDS	IREPBANDn	NLUTSn	IMODE
MONO	B	1	1	1	M, LU	0, 1	B
	INT	8 12 16 32 64	8 12 11-16 17-32 33-64	1		0, 1	B
	R	32	32	1	M	0	B
	C	64	64	1	M	O	B
RGB/LUT	B	1	1	1	LU	3	B
	INT	8	8				
RGB	INT	8 32 64	8 17-32 33-64	3	R, G, B	0	B, P, R, S
YCbCr	INT	8	8	3	Y, Cb, Cr	0	P
MULTI	INT	8 16	8 11-16	1-256	M, R, G, B, LU, or Spaces	0, 1, 3	B, P, R, S
	R	32	32				
1D	INT	8 16 32 64	8 11-16 17-32 33-64	1	Profile Defined	0	B
	C	32 64	32 64				
2D	INT, SI	8 16 32 64	8 11-16 17-32 33-64	2	Profile Defined	0	P
	C	32 64	32 64				
ND	INT, SI	8 16 32 64	8 11-16 17-32 33-64	3-9 0 if "0" XBANDS field is required	Profile Defined	0	P
	C	32 64	32 64				
CMY	INT	8	8	3	C,M,Y	0	B,P,R,S
CMYK	INT	8	8	4	C,M,Y,K	0	B,P,R,S

4.2.6 Symbol segment

4.2.6.1 Symbol subheader

Each symbol segment has a symbol subheader immediately preceding the symbol data. The symbol subheader is used to identify and supply the information necessary to display the symbol data as intended. The format and description for the symbol subheader are shown in Table 6.

Table 6 -- Symbol subheader

FIELD	DESCRIPTION	TYPE	CE/SIZE	VALUE RANGE
SY	BIIF Symbol Subheader This field shall contain the characters SY to identify the subheader as a symbol subheader.	R	A/2	"SY"
SID	Symbol ID This field contains an identification associated with the symbol.	R/PVU	A/10	Profile Defined.
SNAME	Symbol Name This field shall contain a name for the symbol.	R/PVU	U8/20	Profile Defined. Default is spaces.
SSSEC	Symbol Security Profile Specific Parameters This field shall contain profile specific information for the symbol segment security.	R/PVU	U8/167	Profile Defined.
ENCRYP	Encryption Encryption codes and meaning can be defined by profile.	R/PVU	A/1	0=not encrypted. Other codes as specified in profile.
SFMT	Symbol Format: This field shall contain a valid indicator of the representation type of the symbol. A valid value is C, which means Computer Graphics Metafile (CGM). This field identifies the standard as being formatted according to the specified standard. Additional symbol format values may be added through the graphical item registration process.	R/PVV	A/1	"C"=CGM. Other codes specified in profile.
SSTRUCT	Symbol structure parameters This field shall contain profile defined information describing the type and structure of the data (e.g. CGM clear text encoding) in the symbol data field.	R/PVU	A/13	Profile Defined.

Table 6 -- Symbol subheader *(continued)*

FIELD	DESCRIPTION	TYPE	CE/SIZE	VALUE RANGE
SDLVL	Display level This field shall contain a valid value that indicates the graphic display level of the component relative to other BIIF segment. The display level of each BIIF segment (image or symbol) within the BIIF file shall be unique. The image or symbol segment in the BIIF file having the minimum display level shall have attachment level zero.	R	N/3	"001"-"999"
SALVL	Attachment Level This field shall contain a valid value that indicates the attachment level of the component. Valid values for this field are 000 and the display level value of any other image or symbol in the BIIF file. The image or symbol component in the BIIF file having the minimum display level shall have attachment level zero.	R	N/3	"000"-"998"
SLOC	Symbol Location The symbol location is specified by providing the location of the symbol's origin point relative to the position (location) of the item to which it is attached. This field shall contain the symbol location offset from the ILOC or SLOC value of the item to which the symbol is attached or from the origin of the CCS when the symbol is unattached (SALVL=000). A row or column value of 000 indicates no offset. Positive row and column values indicate offsets down and to the right, while negative row and column values indicate offsets up and to the left.	R	N/10	"RRRRRCCCCC" where for positive row and column values RRRRR= "00000"-"99999" and CCCCC= "00000"-"99999"; for negative row and column values RRRRR= "-0001" - "-9999" and CCCCC= "-0001"-"-9999"
SLOC2	Second Symbol Location Default is all zeroes.	R/PVU	N/10	Profile Defined.
SPARMS	Symbol profile specific parameters This field shall contain profile specific information for the symbol such as colour, rotation angle, etc.	R/PVU	A/13	Profile Defined.

Table 6 -- Symbol subheader *(concluded)*

FIELD	DESCRIPTION	TYPE	CE/SIZE	VALUE RANGE
SXSHDL	Extended Subheader Data Length This field shall contain the length in octets of the entire SXSHD field plus three (3) octets. The length is three plus the sum of the lengths of all the Tagged Record Extensions appearing in the SXSHD field. A value of zero shall mean that no Tagged Record Extensions are included in the symbol subheader. If a Tagged Record Extension is too long to fit in the SXSHD field, it shall be put in a data extension segment.	R/PVV	N/5	"00000" or "00003" - "09741"
SXSOFL	Extended Subheader Overflow If present, this field shall contain "000" if the Tagged Record Extensions in SXSHD do not overflow into a DES, or shall contain the sequence number in the file of the DES into which they do overflow. This 3 octet field must be included in the total octet count of SXSHDL. This field shall be omitted if the field SXSHDL contains zero.	C	N/3	000-999
SXSHD	Extended Subheader Data If present, this field shall contain Tagged Record Extensions as allowed by the profile. Tagged record extensions in this field symbol shall contain information pertaining specifically to the symbol. Tagged record extensions shall appear one after the other with no intervening octets. This field shall be omitted if the field SXSHDL contains zero.	C/PVU	length specified by SXSHDL minus "3"	Tagged Record Extensions as allowed by Profile.

4.2.6.2 Symbol data

In the BIIF file, the information describing a symbol segment is represented in a series of adjacent fields grouped into the symbol subheader followed by the symbol data. The field containing the symbol data is called the symbol data field. The symbol data field shall follow immediately after the last field of the corresponding symbol subheader with no intervening special characters to designate the beginning of the symbol data field. Similarly, the symbol subheader of the first symbol shall follow immediately the last octet of data of the last field in the BIIF image section, and the symbol subheader of successive symbols shall follow immediately the last octet of the symbol data field of the preceding symbol. The format of the symbol data field content is specified by the profile. The code "C" in the STYPE field shall designate the use of ISO/IEC 8632, Computer Graphics Metafile (CGM). The BIIF profile shall designate which ISP (s) of CGM are applicable to the specific profile of BIIF. The BIIF profile shall define the mapping from the CGM coordinate space to the BIIF coordinate space. If the designated CGM ISP allows multiple encodings, the BIIF profile may limit the number of encodings actually used.

4.2.7 Text information segment

The text information segment shall be used to store a textual based file or an item of text, such as a word processing file or document. Text items are intended to convey information about the image product contained in the BIIF file. They are not displayed as part of an image.

4.2.7.1 Text subheader

The text subheaders are used to identify and supply the information about the text Information necessary to read and display the text. The format and descriptions for the text subheader are shown in Table 7.

Table 7 -- Text subheader

FIELD	DESCRIPTION	TYPE	CE/SIZE	VALUE RANGE
TE	Text Information Subheader This field shall contain the characters "TE" to identify the subheader as a text subheader.	R	A/2	"TE"
TEXTID	Text ID This field shall contain a valid identification code associated with the text item.	R/PVU	A/10	Profile Defined.
TXTDT	Text Date & Time This field shall contain the UTC (Z) date/time of origination.	R	N/14	CCYYMMDDhhmmss
TXTITL	Text Title This field shall contain the title of the text item.	R/PVU	U8/80	Profile Defined.
TSSEC	Text Security Profile Specific Parameters This field shall contain profile specific information for the text product security and is defined in the profile.	R/PVU	U8/167	Profile Defined.
ENCRYP	Encryption Encryption codes and meaning can be defined by profile, however, 0 = not encrypted.	R/PVU	A/1	0=not encrypted As specified in profile.
TXTFMT	Text Format A three character code indicating the format or template to be used to display the text.	R/PVU	A/3	"STA". "UC2", "UC4", "UT1", UT8". Other codes specified by profile.
TXSHDL	Extended Subheader Data Length This field shall contain the length in octets of the entire TXSHD field plus three (3) octets. A value of zero shall mean that no Tagged Record Extensions are included in the text subheader. If a Tagged Record Extension is too long to fit in the TXSHD field, it shall be put in a data extension segment.	R/PVV	N/5	"00000" or "00003" - "09717"

Table 7 -- Text subheader *(concluded)*

FIELD	DESCRIPTION	TYPE	CE/SIZE	VALUE RANGE
TXSOFL	Extended Subheader Overflow If present, this field shall contain “000” if the Tagged Record Extensions in TXSHD do not overflow into a DES, or shall contain the sequence number in the file of the DES into which they do overflow. This 3 octet field must be included in the total octet count of TXSHDL. This field shall be omitted if the field TXSHDL contains zero.	C	N/3	000-999
TXSHD	Extended Subheader Data If present, this field shall contain Tagged Record Extensions as allowed by the profile. The length of this field shall be the length specified by the field TXSHDL plus three (3). Tagged record extensions in this field for text shall contain information pertaining specifically to the text segment. Tagged record extensions shall appear one after the other with no intervening octets. This field shall be omitted if the field TXSHDL contains zero.	C/PVU	- length specified by TXSHDL minus three (3)	Tagged Record Extensions allowed by profile.

4.2.7.2 Text

The text data field containing the textual item included in the BIIF file shall follow the corresponding text subheader. The text shall consist entirely of the character set specified by the text format code in the subheader (e.g., ISO 10646). The code “STA” in the TXTFMT field shall mean that only BCS characters appear in the text data field. A TXTFMT value of UC2 indicates UCS-2, two octet UCS , UC4 is UCS-4, four octet UCS, UT1 is UTF-1, UCS Transformation Format 1, and UT8 is UTF-8, UCS Transformation Format 8 (Amendment 2 of ISO 10646). Although formatting control characters, CR, LF, FF, are also allowed, these TXTFMT codes indicate that the text characters are otherwise unformatted. The profile identifies the allowed text format codes to be used. Additional codes may be defined by profile to specify formatted text.

4.2.8 Data Extensions

In order to provide complete extensibility to the otherwise fixed format of BIIF, data extensions are available that provide flexibility and versatility. There are three major types of data extensions, Tagged Record Extensions, Data Extension Segments and Reserved Extension Segments.

4.2.8.1 Tagged Record Extensions (TRE): Public and Private.

The Tagged Record Extension is used to extend BIIF by adding additional attributes to existing segments (e.g., IM, SY, TE, see Tables 3, 5, and 6). Although their structure is similar, there are two distinguishing attributes for TREs: the Public TRE and the Private TRE.

Public TREs have identifier names which must be registered by the ISO registration authority. The Private TRE is controlled by the community or group which uses it. All Public TREs have an asterisk “*” in the first location of their identifier name. Private TREs have no asterisk in their name.

The purpose of Private TREs is to allow users of BIIF to establish user defined data constructs within the BIIF structure without the need for international consensus. The private TRE identifier and the structure and content of the user-defined data (TREDATA field) are not coordinated, controlled, nor configuration managed by the ISO or its designated registration authority. Therefore, the use of the TRE is considered to be private in the sense that a specific TRE may only be meaningful to those BIIF users who have mutually agreed to use the TRE in a specified manner.

The following Public TRE usage concepts apply:

— All Public TRE have an asterisk “*” in the first location of its identifier name. The asterisk in the identifier name is the only indicator that a TRE is Public.

- Only those Public TREs accepted and registered by the registration authority shall be used.
- Public TREs shall not be used nor submitted for registration if they adversely impact the utility of the standard features otherwise defined within BIIF and its extensions.
- Nominated Public TREs will be recorded in the ‘Register’ upon approval by the registration authority.
- Upon receipt of a file that contains Public TREs, a BIIF compliant system shall at least ignore the TREs and properly interpret the other legal components of BIIF file.

The following Private TRE usage concepts apply:

- All Private TRE do not have an asterisk “*” in the first location of its identifier name.
- Private TREs are not registered by the ISO registration authority.
- Private TREs should be ignored by all disinterested bodies.
- Upon receipt of a file that contains Private TREs, a BIIF compliant system shall at least ignore the TREs and properly interpret the other legal components of BIIF file.

Each TRE consists of three required fields. These fields are defined in Table 8. A sequence of extensions can appear in the BIIF file header User Defined Data field, UDHD, in any image subheader in its User Defined Image Data field (UDID) and in extended header and subheader data fields (XHD, IXSHD, SXSHD, TXSHD). When the extension carries data associated with the BIIF file as a whole, it should appear in the file header, if sufficient room is available. If the extension carries data associated with an image data item in the BIIF file, it should appear in the UDID or IXSHD field of that item's subheader, if sufficient room is available. Overflow for the TRE may appear in a Data Extension Segment (DES) that is designated to contain extensions. A tagged record extension shall be included in its entirety within the header, subheader or DES selected to contain it. See the Informative Annex E for a sample BIIF File Structure.

Table 8 -- TRE subheader

FIELD	DESCRIPTION	TYPE	CE/SIZE	VALUE RANGE
TRETAG	Unique extension type identifier This field shall contain a valid tag name defined by the profile.	R	A/6	For PUBLIC, 6 BCS-A characters, the first one is an “*”. For PRIVATE, 6 BCS-A characters where the first character shall not be an “*”.
TREL	Length of TREDATA field This field shall contain the length in octets of the data contained in TREDATA.	R	N/5	"00001"-"99985"
TREDATA	User or profile defined data This field shall contain data defined by and formatted according to user or profile specification. The length of this field shall not cause any other BIIF field length limits to be exceeded, but is otherwise fully user-defined.	R	As specified by TREL	For PUBLIC Tagged Record Extensions defined through registration process For PRIVATE TREs User Defined.

4.2.8.2 Data extension segments (DES)

The Data Extension Segment (DES) construct allows for the addition of a variety of differing data types where each type is encapsulated in its own DES. This means that a DES can carry almost any conceivable data type. Potential data types include such things as augmenting imagery with voice annotations, video clip annotations, video/voice annotations, animated graphics, TRE Overflows, Transportable File Structures (TFSs), etc. Each DES carries only one data type, but multiple DESs may be included in a BIIF file, each with the same or differing data type. Each encapsulated extension shall appear in its own Data Extension Segment (DES) and shall conform to the DES structure, Table 9.

The following DES usage concepts apply:

- Only those DES(s) accepted and registered by the registration authority shall be used.
- Upon receipt of a file that contains DES(s), a BIIF compliant system shall at least ignore the DES(s) and properly interpret the other legal components of BIIF file.
- Implementations supporting a specific DES shall at least comply with the minimum conformance requirements specified in the DES description.

The BIIF file header accommodates up to 999 DES. Each DES shall consist of a DES subheader and a DES data field. Within the Data Extension Segment Group in the BIIF file header is found the number of DES(s) in the BIIF file, the length of each DES subheader, and length of the DES data field, DESDATA. The field size specifications in the BIIF file header allows each DES to be just less than one gigabyte in length. The DES subheader shall contain the fields defined in Table 9.

Table 9 -- Data extension segment subheader

FIELD	DESCRIPTION	TYPE	CE/SIZE	VALUE RANGE
DE	Data Extension Subheader This field shall contain the characters "DE" to identify the subheader as a data extension.	R	A/2	"DE"
DESID	Unique DES type identifier This field shall contain a valid identifier as registered with the registration authority.	R	A/25	"TRE_OVERFLOW", or "TRANSPORTABLE_FILE_STRUCT" or as allowed by profile.
DESVER	Version of the data field definition This field shall contain the version number of the use of the tag as registered with the registration authority.	R	N/2	"01-"99"
DESSEC	Security Profile Specific Parameters This field shall contain profile specific information for the DES security and is defined in the profile using the DES.	R	U8/167	As specified by the applicable profile using this DES.
DESOFLW	Overflowed header type This field shall be present if DESID = "TRE_OVERFLOW". Its presence indicates that the DES contains a tagged record extension that would not fit in the BIIF file header or component header where it would ordinarily be located. Its value indicates the data type to which the enclosed tagged record is relevant.	C/PVV	A/6	UDHD, UDID; XHD, IXSHD, SXSHD, TXSHD; otherwise, field is omitted.
DESITEM	Data item overflowed This field shall be present if DESOFLW is present. It shall contain the number of the data item in the BIIF file, of the type indicated in DESOFLW to which the tagged record extensions in the segment apply. For example, if DESOFLW = UDID and DESITEM = 3, then the tagged record extensions in the segment applies to the third image in the BIIF file. If the value of DESOFLW = UDHD or XHD, the value of DESITEM shall be 000.	C	N/3	"000"-"999"
DESSHL	Length of DES-defined subheader fields This field shall contain the number of octets in the field DESSHF. If this field contains 0, DESSHF shall not appear in the DES subheader.	R	N/4	"0000"-"9999"

Table 9 -- Data extension segment subheader *(concluded)*

FIELD	DESCRIPTION	TYPE	CE/SIZE	VALUE RANGE
DESSHF	DES-defined subheader fields This field shall contain DES-defined subheader fields. Data in this field shall be formatted according to the registered DES specification.	C/PVU	Value specified in DESSHL	Defined through registration process.
DESDATA	DES-defined data field This field shall contain data of either binary or character types defined by and formatted according to the user's specification. However, if the DESID is "TRE_OVERFLOW," the tagged records shall appear according to their definition with no intervening octets. If the DESID is "TRANSPORTABLE_FILE_STRUCT" then the format is specified in Annex A. The length of this field shall not cause any other BIIF field length limits to be exceeded, but is otherwise fully user-defined.	R/PVU	Per value from file header	Defined through registration process.

4.2.8.3 Defined DESs

There are two currently defined DES type identifiers: "TRE_OVERFLOW" and "TRANSPORTABLE_FILE_STRUCT".

4.2.8.3.1 TRE_OVERFLOW DES

The "TRE_OVERFLOW" is used when a series of TRE extensions appears in a DES as "overflow" from BIIF header or any subheader. A separate DES is used for each header or subheader field that overflows. Which header or subheader field overflowed is indicated in the DESOFLOW and DESITEM field contents. The DES for encapsulating Tagged Record Extensions which overflow from the file header or standard segment subheaders is defined in Table 10.

Table 10 -- Data extension segment subheader for TRE OVERFLOW

FIELD	DESCRIPTION	TYPE	CE/SIZE	VALUE RANGE
DE	Data Extension Subheader This field shall contain the characters "DE" to identify the subheader as a data extension.	R	A/2	"DE"
DESID	Unique DES type identifier.	R	A/25	"TRE_OVERFLOW"
DESVER	Version of the data field definition This field shall contain the version number of the use of the tag as registered with the registration authority.	R	N/2	"01"
DESCLAS	Security Profile Specific Parameters This field shall contain profile specific information for the DES security. and is defined in the profile using this DES.	R	U8/167	As specified by the applicable profile using this DES.

Table 10 -- Data extension segment subheader for TRE_OVERFLOW *(concluded)*

FIELD	DESCRIPTION	TYPE	CE/SIZE	VALUE RANGE
DESOFLW	Overflowed header type This field shall be present if DESTAG = "TRE_OVERFLOW". Its presence indicates that the DES contains a TRE that would not fit in the BIIF file header or component header where it would ordinarily be located. Its value indicates the data type to which the enclosed tagged record is relevant.	C	A/6	UDHD, UDID, XHD, IXSHD, SXSHD, TXSHD; otherwise, field is omitted.
DESITEM	Data item overflowed This field shall be present if DESOFLW is present. It shall contain the number of the data item in the BIIF file, of the type indicated in DESOFLW to which the Tagged Record Extensions in the segment apply. If the value of DESOFLW = UDHD or XHD, the value of DESITEM shall be 000.	C	N/3	"000"-"999"
DESSHL	Length of DES-defined subheader fields.	R	N/4	"0000"
DESDATA	DES-defined data field This field shall contain data of either binary or character types defined by and formatted according to the user's specification. The length of this field shall not cause any other BIIF field length limits to be exceeded, but is otherwise fully user-defined.	R	Determined by User, Profile Defined	Tagged Record Extensions with no intervening octets.

4.2.8.3.2 TRANSPORTABLE_FILE_STRUCT DES

The TRANSPORTABLE_FILE_STRUCT DES is used to contain TFS Commands as specified in Annex A. This DES is defined in Table 11.

Table 11 -- Data extension segment subheader for TRANSPORTABLE_FILE_STRUCT

FIELD	DESCRIPTION	TYPE	CE/SIZE	VALUE RANGE
DE	Data Extension Subheader	R	A/2	"DE"
DESID	Unique DES type identifier	R	A/25	"TRANSPORTABLE_FILE_STRUCT"
DESVER	Version of the data field definition This field shall contain the version number of the use of the tag as registered with the registration authority.	R	N/2	"01"
DESCLAS	Security Profile Specific Parameters This field shall contain profile specific information for the DES security and is defined by the profile using the DES.	R	U8/167	As specified by the applicable profile using this DES.

Table 11 -- Data extension segment subheader for TRANSPORTABLE_FILE_STRUCT *(concluded)*

FIELD	DESCRIPTION	TYPE	CE/SIZE	VALUE RANGE
DESSHL	Length of DES-defined subheader fields	R	N/4	"0000
DESDATA	DES-defined data field This field shall contain data of either binary or character types defined by and formatted according to the user's specification. The length of this field shall not cause any other BIIF field length limits to be exceeded, but is otherwise fully user-defined.	R	Per value from file header	TFS Commands as specified in Annex A.

4.2.8.4 Reserved extension segments (RES)

The RES construct provides the same mechanism as the DES construct for adding a variety of new data types for inclusion in BIIF files. No use of the RES construct is currently specified in this standard. The RES is reserved for data types that need to be placed at or near the end of the file. For example, a digital signature that covered the whole file could be defined for placement in a RES to verify the bit level integrity of BIIF file. The RES subheader shall contain the fields defined in Table 12.

The following RES usage concepts apply:

- Only those RES(s) accepted and registered by the registration authority shall be used.
- Upon receipt of a file that contains RES(s), a BIIF compliant system shall at least ignore the RES(s) and properly interpret the other legal components of BIIF file.
- Implementations supporting a specific RES shall at least comply with the minimum conformance requirements specified in the RES description.

Table 12 -- Reserved extension segment subheader

FIELD	DESCRIPTION	TYPE	CE/SIZE	VALUE RANGE
RE	Reserved Extension Subheader This field shall contain the characters "RE" to identify the subheader as a data extension.	R	A/2	“RE”
RESID	Unique RES type identifier This field shall contain a valid identifier as defined through registration.	R	A/25	Unique Identifier as registered with ISO.
RESVER	Version of the data field definition This field shall contain the version number of the use of the tag as defined by the registration process.	R	N/2	"01"-"99"
RESSEC	Security Specific Parameters This field shall contain specific information for the product security as defined in the profile being used.	R	U8/167	As specified for the applicable profile using the RES.
RESSHL	Length of registration-defined subheader fields This field shall contain the number of octets in the field RESSHF. If this field contains 0, RESSHF shall not appear in the RES subheader.	R	N/4	"0000"-"9999"
RESSHF	Registration-defined subheader fields This field shall contain registration-defined subheader fields. Data in this field shall be formatted according to registered specification.	C/PVU	Value specified in RESSHL	Defined through registration.
RESDATA	Registration-defined data field This field shall contain data of either binary or character types defined by and formatted according to the user's specification.	R/PVU	Per value from file header	Defined through registration.

5 Conformance profiles and extensions

ISO/IEC 12087-5 (BIIF) utilizes the concept of conformance profiles as established by ISO/IEC 12087-1 and International Standardized Profiles (ISPs) as established by ISO/IEC TR 10000-1 Third Edition, 1995-12-15. Registering an ISP is the standardized means for tailoring BIIF for use by communities of interest that have different functional scopes in order to suit a variety of user requirements. Since BIIF is very flexible, it has many options; the use of which must be constrained for implementation if file exchange interoperability is to be achieved within a designated community of interest. The BIIF profiling approach allows specification of a hierarchical (nested) set of implementation variables (complexity levels) within a specific ISP. The BIIF profile allows inclusion of data types defined by external content profiles (profiles of ISO/IEC 12087, registered ISO profiles external to 12087, and other approved standards documents and registered items). Finally, the BIIF profile provides additional means of extensibility through the registration of tagged and encapsulated extensions.

5.1 Profiles

A means for profiling is provided as a normative annex to this standard (Annex C). This annex consists of a set of proforma tables to be used as templates for specifying BIIF ISPs. A representative use of the proforma is shown as the Model Profile.

New ISPs nominated for registration can be developed by referring to the Model Profile as the starting point and simply identifying additional capabilities and their constraints. Although the inclusion of the Model Profile within registered profiles is not mandatory, it is considered a minimally conformant use of BIIF. Inclusion of the Model Profile in other profiles promotes an increased potential for a basic level of interoperability and data portability among implementations of differing BIIF profiles. This basic level of interoperability can be achieved by providing an implementation option which allows the user to limit the content of BIIF files to the constraints of the model profile.

When developing a new profile for a specific application domain, the Model Profile, existing profiles, and referenced content profiles (such as those for PIKS, CGM, JPEG, etc.) should be examined to determine if these meet the requirements for the targeted application domain. The Model Profile may be used as the starting point for the new profile, or other profiles that have been developed for communities with similar interests may be used.

5.2 Profile specific header/subheader dependencies proforma

The standard describes a number of header and sub-header fields, the formats of which are subject to profile specific definition and constraints. Table C.1 through C.10 in Annex C provides a proforma to assist in the submission of the formally registered profiles of this standard. A completed proforma, once validated and approved by the registration authority, becomes the definition of profile specific constraints to be used by implementers of the profile.

5.3 Complexity level proforma

Complexity levels within a functional profile provide a means to define a set of nested features and constraints within a specific profile. Having these nested feature sets (where the restrictions on the set of attributes are expected to vary in progression and only for specified values) increases the potential for a high degree of interoperability between disparate implementations having various levels of computational resources (e.g. laptops, desktops, workstations, super-computers, etc.).

A "complexity level" is defined by a nested set of Profile Variants (PVs) that are a proven combination that work together. The same set of profile variants is in effect across the range of complexity levels with the general framework of support remaining the same for all.

The rules for developing complexity levels within a single Functional Profile are:

- Select the set of Profile Variants from those permitted by BIIF standard (Clause 4 tables indicate attributes that are considered to be potential profile variants);
- Specify the permitted values (minimum to maximum, or from an enumerated set or special syntax and semantics as defined by the profile) for each PV at each complexity level.
- All other data conformance requirements (the "framework") remain the same across the complexity level progression.

The proformas presented in Annex C provide a standard way of expressing the selections.

5.4 Implementation support requirements

5.4.1 General support requirements

There are three primary implementation support considerations in establishing compliance with BIIF:

— Compliance of individual files with the selected BIIF profile;
— Ability to produce BIIF compliant files within profile constraints; and,
— Ability to properly interpret and present BIIF compliant files.

The above clauses describe aspects of BIIF profiling that address the first consideration above, the compliance of the file format itself. This is the primary purpose of this standard. However, the BIIF profiling approach allows for the profile proponent to identify implementation support requirements applicable to the concept of operation within the application domain envisioned to use BIIF profile. Table C.10 in Annex C provides a proforma for inclusion of implementation support requirements as part of a registered profile; the format is free text.

Functional areas to be addressed should include at a minimum, the following:

— Image reproductive fidelity (e.g., colour fidelity, requirements for mensuration, font fidelity of shape and size, level of geometric precision, etc.);
— Compression goals and related quality of image;
— Any special content transforms that may be needed to correctly interpret data (e.g., the mathematical algorithms for converting from a non-standard colour model to a standard model); and,
— Any special handling or processing rules that provide application semantics (e.g., the special interpretation of a hyperlink, a user interface function, a timing consideration, or constraint).

5.4.2 Producing and interpreting BIIF files

An implementation of BIIF may be capable of producing BIIF files, interpreting BIIF files, or both.

5.4.2.1 Producing BIIF files

In general, a BIIF profile compliant implementation does not need to be capable of producing BIIF files that exercise every option and permutation of features allowed within the specific profile. All files produced must be within (i.e. not violate) the constraints of the applicable profile. The specification of implementation support requirements may impose additional conditions or constraints on implementations that produce BIIF files.

5.4.2.2 Interpreting BIIF files

A BIIF profile compliant interpret implementation shall be able to interpret and use any combination and permutation of BIIF file format options within the constraints of the applicable profile. The specification of implementation support requirements may offer additional clarification, but shall not violate these principles.

5.5 Defined extensions

Extensibility of BIIF is supported by the registration and use of Tagged Record Extensions (TREs), Data Extension Segments (DESs), and Reserved Extension Segments (RESs) as defined in Clause 4. The registration process is that defined by ISO/IEC 9973, Procedures for Registration of Graphical Items.

5.6 Profile registration

Alternative profiles of BIIF may be created and registered for general usage. These profiles shall follow the model profile proforma shown in Annex B. The model profile lists all the required components, of which selections, restrictions, and qualifications may be made. These can include restrictions on options and fields.

Profiles may reference standardized portions of the standard, registered extensions of the standard, or other ISO standards and ISPs. Profiles shall not contain additional functionality. Additional content should be registered separately.

The registration process for ISPs generally involves the following steps as detailed in ISO/IEC TR10000-1:

a) Develop the profile per the Model Profile

b) Submit the profile to the Registration Authority
c) Review of the profile by the Registration Authority
d) If the Registration Authority approves, skip to step f
e) If the Registration authority makes a request for revision, return to step a
f) International ballot of profile
g) If there is International approval, the profile becomes available from ISO & Registration Authority
h) If there is International denial, return to step a.

Annex A
(normative)

Transportable file structure

A.1 Transportable file structure (TFS)

The Transportable File Structure (TFS) Data Extension Element allows for configuration, data request, commands, and PIKS object data to be stored in hierarchy order with Metadata associated for each level. The TFS shall either contain data for the objects or provide a computer system with an unambiguous reference to the data. PIKS objects can be stored in a TFS to perform the required image processing steps to a given image. While the TFS has the capability to express complicated structure relating several hierarchy objects, the TFS can relate a BIIF image with several PIKS objects in a simple expression. The TFS can be parsed using a state machine. Major sections of the TFS are delimited with TFS Delimiter commands. The initial state of the TFS is the TFS state. The TFS Descriptor commands describing the overall structure of the TFS, the TFS version Metadata, security, subscription, configuration, and an index into the TFS transports are found in this state. The next state of the TFS is the Transport State. The Transport Descriptor commands describe the security, Metadata, and index into the transport body components. Within the transport body the TFS enters one or more Transport Profile states. The Transport Profile Descriptor commands describe the security, Metadata, and index for each Transport Profile state. The Transport Profile body contains objects, actions, or other nested Transport Profiles. Transport Profile objects contain data formatted according to object type. The object data may contain an unambiguous reference to data stored elsewhere, such as in the BIIF file, or contain the actual object data. More than one Transport Profile can be contained within a transport body and the TFS can contain more than one transport. Furthermore, the Transport Profiles can be nested to create hierarchical structures. For the simple case of one BIIF image and several PIKS objects, these objects should be inside a single Transport Profile for processing. Each TFS, transport, and Transport Profile state contains security, Metadata, and an associated index into the next hierarchical level. The TFS escape state provides a mechanism for the insertion of application defined data. The TFS escape state terminates upon completion of the TFS escape command. All other states have a begin state and an end state.

A.1.1 TFS Commands

The following describes TFS commands grouped by command class. The delimiter commands change the TFS state. The descriptor commands provide data within each state.

1. TFS Delimiter Commands
   - BEGIN TFS
   - BEGIN TRANSPORT
   - BEGIN TRANSPORT BODY
   - BEGIN TRANSPORT PROFILE
   - BEGIN TRANSPORT PROFILE BODY
   - END TRANSPORT PROFILE
   - END TRANSPORT
   - END TFS

2. TFS Descriptor Commands
   - TFS VERSION
   - TFS SECURITY
   - TFS METADATA
   - TFS INDEX
   - TFS SUBSCRIPTION
   - TFS CONFIGURATION
   - TFS CONFIG DATA

3. Transport Descriptor Commands
   - TRANSPORT SECURITY
   - TRANSPORT METADATA
   - TRANSPORT INDEX

4. Transport Profile Descriptor Commands
   - TRANSPORT PROFILE SECURITY
   - TRANSPORT PROFILE METADATA
   - TRANSPORT PROFILE INDEX

5. Transport Profile Commands
   - TRANSPORT PROFILE ACTION
   - TRANSPORT PROFILE OBJECT

6. TFS User Commands
   TFS ESCAPE

A.1.2 TFS encoding

Each TFS command is identified by the combination of the TFS State (TS) and TFS Substate (TSS) fields which comprise the first two bytes of the command. The identification is followed by two or four bytes called the Command Parameter Length (CPL) specifying the length of the balance of the command. The balance of the command is a series of parameters whose content depends upon the identity of the command. The TFS commands are encoded in binary form. The 8 most significant bits (MSB) of the first word identify the TFS state in which the command belongs. The 8 least significant bits (LSB) specify the TFS substate of the command. The next 2 bytes are interpreted as a signed positive integer containing the length or the CPL of the command parameters. Parameters are padded with a trailing ASCII null byte (0) to ensure that all subsequent commands begin on a 16-bit word boundary. The trailing null byte of the last parameter is not included in the parameter list length. If the CPL is greater than 32,767, then the CPL shall be two 16-bit words long with the MSB of the high order byte (word 1) set to 1. The value of the CPL is that of the remaining 31 bits in the two words and can contain values up to 999,999,999 bytes for the full CPL. Note, the TFS size must comply with the overall size of the DES data field in which it is included. Field numbers in tables contained in this annex are in hexadecimal notation.

Table A.1 -- Encoding of a TFS command

MSB 15	14	13	12	11	10	9	8	7	6	5	4	3	2	LSB 1	0
TFS State (TS)								TFS Substate (TSS)							
Command Parameter Length (CPL)															
command parameters															

A.1.3 TFS command flow

The following flow diagram displays all the TFS commands contained in a Metafile. The commands are executed in sequential order. The TFS Descriptor Commands occur after the BEGIN TFS command and before the BEGIN TRANSPORT command. The Transport Body Descriptor Commands occur after the BEGIN TRANSPORT command and before the BEGIN TRANSPORT BODY command. The Transport Profile Descriptor Commands occur after the BEGIN TRANSPORT PROFILE command and before the BEGIN TRANSPORT PROFILE BODY command. The Transport Profile Commands occur after the BEGIN TRANSPORT PROFILE BODY command and before the END TRANSPORT PROFILE command. BEGIN TRANSPORT PROFILE and associated commands can be nested. The TFS ESCAPE command can occur at any position in the TFS where any other command can occur except the BEGIN TFS and END TFS commands. The commands enclosed within matching brackets are optional within a given metafile.

```
BEGIN TFS
  TFS VERSION
  TFS SECURITY
  [TFS INDEX]
  [TFS SUBSCRIPTION]
  [TFS CONFIGURATION]
  [TFS CONFIG DATA]
  [TFS METADATA]
  [BEGIN TRANSPORT
        TRANSPORT SECURITY
        [TRANSPORT METADATA]
        [TRANSPORT INDEX]
        [BEGIN TRANSPORT BODY
              [BEGIN TRANSPORT PROFILE
                    TRANSPORT PROFILE SECURITY
                    [TRANSPORT PROFILE METADATA]
                    [TRANSPORT PROFILE INDEX]
                    [BEGIN TRANSPORT PROFILE BODY
```

```
                              [TRANSPORT PROFILE ACTION
                              [TRANSPORT PROFILE OBJECT]
                              [BEGIN TRANSPORT PROFILE
                                  ...
                              END TRANSPORT PROFILE]
               END TRANSPORT PROFILE]]
    END TRANSPORT]
  END TFS
```

A.2 TFS command formats

The following subclauses list the required TFS commands along with the encoding.

A.2.1 TFS delimiter commands

The TFS Delimiter commands define boundaries for significant structures within the TFS. The following table defines the TS, TSS, and the CPL for the TFS Delimiter commands. The value x for CPL indicates that a non-zero value shall be used if there is data supplied with the command.

Table A.2 -- TFS delimiter command table

TFS COMMAND NAME	TS	TSS	CPL
BEGIN TFS	10	1	x
BEGIN TRANSPORT	10	2	x
BEGIN TRANSPORT BODY	10	3	0
BEGIN TRANSPORT PROFILE	10	4	x
BEGIN TRANSPORT PROFILE BODY	10	5	0
END TRANSPORT PROFILE	10	6	0
END TRANSPORT	10	7	0
END TFS	10	8	0

A.2.1.1 Begin TFS command

The BEGIN TFS command shall have non-zero values for CPL if it has parameter data. In this case, the parameter is a name associated with the TFS file. This name is represented using the character string C1, C2, ... Cn with length n. The following tables describes the even and odd format with x = n+1;0. No data will be in the length or character fields of the command format as depicted in Tables A.3 and A.4 when n and CPL equal 0.

Table A.3 -- TFS command even form

MSB 15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0 LSB
TS	TSS
CPL = n+1;0	
length = n (even)	C1
C2	. . .
Cn	0

Table A.4 -- TFS command odd form

MSB 15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0 LSB
TS	TSS
CPL = n+1;0	
length = n (odd)	C1
C2	. . .
C(n-1)	Cn

A.2.1.2 Begin transport command

The BEGIN TRANSPORT command shall have non-zero values for CPL if it has parameter data. In this case, the parameter is a name associated with the receiver of the transport data. This name is represented using the character string C1, C2, ... Cn with length n and conforms to the format depicted in Tables A.3 and A.4 with associated TS and TSS and non-zero CPL.

A.2.1.3 Begin transport body command

The BEGIN TRANSPORT BODY command conforms to the format depicted in Tables A.3 and A.4 with associated TS and TSS and zero CPL. See Tables A.3 and A.4.

A.2.1.4 Begin transport profile command

The BEGIN TRANSPORT PROFILE command shall have non-zero values for CPL if it has parameter data. In this case, the parameter is a name associated with the Transport Profile. This name is represented using the character string C1, C2, ... Cn with length n and conforms to the same format as the Begin TFS command with associated TS and TSS and non-zero CPL. See Tables A.3 and A.4.

A.2.1.5 Begin transport profile body command

The BEGIN TRANSPORT PROFILE BODY command conforms to the same format as the Begin TFS command with associated TS and TSS and zero CPL.

A.2.1.6 End transport profile command

The END TRANSPORT PROFILE command conforms to the same format as the Begin TFS command with associated TS and TSS and zero CPL.

A.2.1.7 End transport command

The END TRANSPORT command conforms to the same format as the Begin TFS command with associated TS and TSS and zero CPL.

A.2.1.8 End TFS command

The END TFS command conforms to the same format as the Begin TFS command with associated TS and TSS and zero CPL.

A.2.2 TFS descriptor commands

The TFS Descriptor commands describe the functional content, security, and characteristics of the TFS. The following table defines the TS, TSS, and the CPL for the TFS Descriptor commands. The value x for CPL indicates that a non-zero value shall be used if there is data supplied with the command. The value 2 for CPL indicates that the CPL value is 2.

Table A.5 -- TFS descriptor command table

TFS COMMAND NAME	TS	TSS	CPL
TFS VERSION	20	1	2
TFS SECURITY	20	2	x
TFS SUBSCRIPTION	20	3	x
TFS CONFIGURATION	20	4	2
TFS CONFIG DATA	20	5	x
TFS METADATA	20	6	x
TFS INDEX	20	7	x

A.2.2.1 TFS version command

The following is the TFS VERSION command format. The VERSION shall be an integer.

Table A.6 -- TFS version

MSB 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	LSB 0
TS								TSS							
CPL															
VERSION															

A.2.2.2 TFS security command

The TFS SECURITY command conforms to the same format as the TFS Delimiter commands with associated TS and TSS and non-zero CPL.

A.2.2.3 TFS subscription command

The following is the TFS SUBSCRIPTION command format. The SUBSCRIPTION value is 1 to subscribe to data objects, 2 for a data pull, and 3 to cancel subscription. Each subscription service consists of an unsigned 8-bit service name length followed by a character name. Each character name shall be in the form "name = identifier".

Table A.7 -- TFS subscription command

<table>
<tr><td>MSB
15 14 13 12 11 10 9 8</td><td>LSB
7 6 5 4 3 2 1 0</td></tr>
<tr><td>TS</td><td>TSS</td></tr>
<tr><td colspan="2">CPL</td></tr>
<tr><td colspan="2">SUBSCRIPTION</td></tr>
<tr><td>length = a</td><td>C1</td></tr>
<tr><td>C2</td><td>. . .</td></tr>
<tr><td>Ca</td><td>. . .</td></tr>
<tr><td>length = b</td><td>C1</td></tr>
<tr><td>C2</td><td>. . .</td></tr>
<tr><td>Cb</td><td>. . .</td></tr>
</table>

The following table describes the contents for the TFS Subscription command. Each command is of the form "name = identifier". The URL identifier determines how to send the TFS to the subscriber. The OBJECT-NAME identifier determines how the TFS objects are to be stored.

Table A.8 -- TFS subscription contents table

NAME	IDENTIFIER
USERNAME	Login Name
PASSWORD	Login Password
HOSTNAME	Computer Host Name or IP Address
URL	URL identifier - What transfer to use to deliver the TFS (E-MAIL, FTP, HTTP, ASYNCHRONOUS, etc.)
DIRECTORY	Directory where TFS is to be deposited
FILENAME	File name to use when depositing the TFS in above directory
OBJECT_NAME	How object is to be stored 0=METADATA ONLY - No object data 1=TFS - object data stored in TFS 2 =URL - TFS string point to object data

A.2.2.4 TFS configuration command

The following is the TFS CONFIGURATION command format. The CONFIGURATION value is 0 to provide configuration and 1 to request configuration. Note, if the CONFIGURATION value is 0 then the TFS CONFIG DATA command is required.

Table A.9 -- TFS configuration command

MSB 15 … 8	7 … 0 LSB
TS	TSS
CPL	
CONFIGURATION	

A.2.2.5 TFS configuration data command

The following is the TFS CONFIG DATA command format. Each configuration service consists of an unsigned 8-bit service name length followed by the configuration service name. This command is supplied by a subscriber when the TFS Configuration Command CONFIGURATION value 0 and supplied to a subscriber upon request. Each configuration service name shall be in the form "OBJECT_NAME=service" where OBJECT_NAME is a valid object name to be used in the TFS and service is the required program to execute for that object.

Table A.10 -- TFS config data command

MSB 15 … 8	7 … 0 LSB
TS	TSS
CPL	
length = a	C1
C2	. . .
Ca	. . .
length = b	C1
C2	. . .
Cb	. . .

A.2.2.6 TFS metadata command

The TFS METADATA command conforms to the same format as the TFS Delimiter commands with associated TS and TSS and non-zero CPL. See Tables A.3 and A.4.

A.2.2.7 TFS index command

The TFS INDEX command provides a quick entry into the Transports of the TFS. The NUMBER_OF_INDEXES field is an integer field that specifies the number of indexes, one for each BEGIN TRANSPORT command. Each index consists of an unsigned 8-bit index name length followed by the character index name followed by the index. The name must match the name contained in the BEGIN TRANSPORT parameter to which the index points. The index offset is expressed in bytes and is calculated relative to the BEGIN TFS command (beginning of the TFS) whose first byte has a value of 1 to the BEGIN TRANSPORT command. There must be an entry for each BEGIN TRANSPORT command in the TFS and the index must be to and inclusive of the first byte of those commands.

Table A.11 -- TFS index command

MSB 15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 LSB 0
TS	TSS
CPL	
NUMBER_OF_INDEXES	
length = a	C1
C2	. . .
Ca	. . .
LONG_OFFSET_WORD_1	
LONG_OFFSET_WORD_2	
length = b	C1
C2	. . .
Cb	. . .
LONG_OFFSET_WORD_1	
LONG_OFFSET_WORD_2	
...	

A.2.3 Transport descriptor commands

The Transport Body Descriptor Commands describe the functional content, security, and characteristic of the transport body. The following table defines the TS, TSS, and the CPL for the Transport Descriptor commands. The value x for CPL indicates that a non-zero value shall be used if there is data supplied with the command.

Table A.12 -- Transport descriptor command table

TFS COMMAND NAME	TS	TSS	CPL
TRANSPORT SECURITY	30	1	x
TRANSPORT METADATA	30	2	x
TRANSPORT INDEX	30	3	x

A.2.3.1 Transport security command

The TRANSPORT SECURITY command conforms to the same format as the TFS Delimiter commands with associated TS and TSS and non-zero CPL.

A.2.3.2 Transport metadata command

The TRANSPORT METADATA command conforms to the same format as the TFS Delimiter commands with associated TS and TSS and non-zero CPL. See Tables A.3 and A.4.

A.2.3.3 Transport index command

The TRANSPORT INDEX command provides a quick entry into the results contained in the Transport Body. The TRANSPORT INDEX command conforms to the same format as the TFS INDEX command with associated TS, TSS, and non-zero CPL. The NUMBER_OF_INDEXES field is an integer field that specifies the number of indexes, one for each BEGIN_TRANSPORT PROFILE command. Each index consists of an unsigned 8-bit index name length followed by the character index name followed by the index. The name must match the name contained in the BEGIN TRANSPORT PROFILE parameter to which the index points. The index offset is expressed in bytes and is calculated relative to the BEGIN TRANSPORT command whose first byte has a value of 1. There must be an entry for each BEGIN TRANSPORT PROFILE command in the Transport Body, and the index must be to and inclusive of the first byte of those commands. See Table A.11.

A.2.4 Transport Profile descriptor commands

The Transport Profile Descriptor commands describe the functional content, security, and characteristic of the Transport Profile. The following table defines the TS, TSS, and the CPL for the Transport Profile Descriptor commands. The value x for CPL indicates that a non-zero value shall be used if there is data supplied with the command.

Table A.13 -- Transport profile descriptor command table

TFS COMMAND NAME	TS	TSS	CPL
TRANSPORT PROFILE SECURITY	40	1	x
TRANSPORT PROFILE METADATA	40	2	x
TRANSPORT PROFILE INDEX	40	3	x

A.2.4.1 Transport Profile security command

The TRANSPORT PROFILE SECURITY command conforms to the same format as the TFS Delimiter commands with associated TS and TSS and non-zero CPL. See Tables A.3 and A.4.

A.2.4.2 Transport profile metadata command

The TRANSPORT PROFILE METADATA command conforms to the same format as the TFS Delimiter commands with associated TS and TSS and non-zero CPL. See Tables A.3 and A.4.

A.2.4.3 Transport profile index command

The TRANSPORT PROFILE INDEX command provides a quick entry into the Actions, Objects, and embedded Transport Profiles contained in the Transport Profile Body. The TRANSPORT PROFILE INDEX command conforms to the same format as the TFS INDEX command with associated TS, TSS, and non-zero CPL. The NUMBER_OF_INDEXES field is an integer field that specifies the number of indexes, one for each TRANSPORT PROFILE ACTION, TRANSPORT PROFILE OBJECT, and BEGIN TRANSPORT PROFILE command. Each index consists of an unsigned 8-bit index name length followed by the character index name followed by the index. The name must match the name contained in the TRANSPORT PROFILE ACTION, TRANSPORT PROFILE OBJECT, or BEGIN TRANSPORT PROFILE parameters to which the index points. The index offset is expressed in bytes and is calculated relative to the BEGIN TRANSPORT PROFILE command whose first byte has a value of 1. There must be an entry for each TRANSPORT PROFILE COMMAND, TRANSPORT PROFILE OBJECT, and BEGIN TRANSPORT PROFILE commands in the Transport Profile Body, and the index must be to and inclusive of the first byte of those commands. See Table A.11.

A.2.5 Transport profile commands

The Transport Profile commands describe the individual objects contained in each Transport Profile. The following table defines the TS, TSS, and the CPL for the Transport Profile commands. The value x for CPL indicates that a non-zero value shall be used if there is data supplied with the command.

Table A.14 -- Transport profile command table

TFS COMMAND NAME	TS	TSS	CPL
TRANSPORT PROFILE ACTION	50	1	x
TRANSPORT PROFILE OBJECT	50	2	x

A.2.5.1 Transport profile action command

The TRANSPORT PROFILE ACTION command conforms to the same format as the TFS Delimiter commands with associated TS and TSS and non-zero CPL. See Tables A.3 and A.4. The command shall be in the form "OBJECT_NAME=value" where command_name and value are defined by the Transport Profile.

A.2.5.2 Transport profile object command

The TRANSPORT PROFILE OBJECT command provides a mechanism to reference all-source objects inside a Transport Profile for processing. Each object has a type followed by a name and then the specified data or a pointer to the data. The following is the TRANSPORT PROFILE OBJECT command format. The OBJECT_TYPE field is an integer that specifies the type of data. If OBJECT_TYPE=1, then the object storage is given by the TFS Configuration Data command by the configuration service "OBJECT_NAME=action_storage". If OBJECT_TYPE=2, then the object data is contained as a BIIF component, and that component number is given in the OBJECT_DATA field. The next field is the length of the OBJECT_NAME followed by the OBJECT_NAME in characters (C1, C2, ..., Ca). The OBJECT_STORAGE field is an integer that determines how the data is stored. If OBJECT_STORAGE=0 then there is metadata only and no object data is stored in OBJECT_DATA. . If OBJECT_STORAGE does not equal 0, then object data is stored in the TFS or BIIF with the TFS string pointing to object data.

Table A.15 -- Transport profile object command

MSB 15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0 LSB
TS	TSS
CPL	
OBJECT_TYPE	
length = a	C1
C2	. . .
Ca	. . .
OBJECT_STORAGE	
OBJECT_DATA	
...	

The following table describes the contents of the OBJECT_DATA field for the OBJECT_TYPE=2 (BIIF segment). The first entry is the name of the BIIF segment identifier (IM, SY, TE, DE, or RE) concatenated with the segment name (IID, SID, TEXTID, DESTAG, or RESTAG). The second entry is the integer value for the BIIF_SEGMENT. The BIIF_SEGMENT points to the associated Display Level (1-999) in the BIIF file for images and symbols. When the BIIF_SEGMENT is a TEXT, DES, or a RES segment, the BIIF_SEGMENT is the sequence number of occurrence for the given standard segment type.

Table A.16 -- Object data for BIIF transport profile object

MSB 15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0 LSB
length = a	C1
C2	. . .
Ca	. . .
BIIF_SEGMENT	

The following table describes the contents for the OBJECT_STORAGE and associated OBJECT_DATA fields for the TRANSPORT PROFILE OBJECT command.

Table A.17 -- Object storage and object data format

OBJECT_STORAGE	OBJECT_DATA (FORMAT)
0=METADATA ONLY	No object data
1=TFS	Object data stored in TFS or BIIF
2=URL	TFS string pointing to object data

A.2.5.2.1 Transport Profile object command parameters for PIKS objects

The TFS provides the capability to include PIKS Objects for image processing. PIKS Objects are associated with an image using the BIIF_SEGMENT as defined in clause A.2.5.2, Transport Profile object command. Both the Transport Profile object command containing the BIIF_SEGMENT and the Transport Profile containing the PIKS Objects are embedded inside a single Transport Profile to define the relationship. The following table defines the minimum size for the PIKS data types used for PIKS objects.

Table A.18 – PIKS Objects minimum size

DATATYPE	MINIMUM SIZE IN BITS	C DATATYPE (INFORMATIVE)
BP	8	UNSIGNED CHAR
NP	64	UNSIGNED LONG INT
SP	32	INT
RP	32	FLOAT
CP	64	FLOAT PAIR
CS	8	CHAR
BD	1	UNSIGNED CHAR
ND	8	CHAR
SD	16	SHORT
RD	32	FLOAT
CD	64	FLOAT PAIR

The following table defines the OBJECT_DATA field contents for the OBJECT_TYPE of PIKS. The name of the PIKS object shall be presented as the first parameter as a character string. The second parameter shall be a character string specifying th e object identifier. The following parameters and lengths are defined in the table below.

Table A.19 -- PIKS objects data structures

PIKS OBJECT	Parameters	Length
OBJECT_CLASS “PIKS_IMAGE”	CS	10
PIKS_IMAGE_ID	CS	Max 32
X_SIZE	NP	1
Y_SIZE	NP	1
Z_SIZE	NP	1
T_SIZE	NP	1
B_SIZE	NP	1
X_BANDTYPE	NP	1
Y_BANDTYPE	NP	1
Z_BANDTYPE	NP	1
T_BANDTYPE	NP	1
B_BANDTYPE	NP	1
X_WHITE_POINT	RP	1
Y_WHITE_POINT	RP	1
Z_WHITE_POINT	RP	1
STRUCTURE	SP	1
COLOUR_SPACE	SP	1
DATA Note - For this object the image data is located in the image segment having this display level identified in this BIIF_ATTACHMENT_LEVEL associated with this TFS.	BD,ND,SD,RD,CD	[X_SIZE]*[SIZEOF(X_BANDTYPE)]* [Y_SIZE]*[SIZEOF(Y_BANDDYPE)]* [Z_SIZE]*[SIZEOF(Z_BANDTYPE)]* [T_SIZE]*[SIZEOF(T_BANDTYPE)]* [B_SIZE]*[SIZEOF(B_BANDTYPE)]

Table A.19 -- PIKS objects data structures *(continued)*

PIKS OBJECT	Parameters	Length
OBJECT_CLASS “PIKS_HISTOGRAM”	CS	14
HISTOGRAM_ID	CS	Max 32
SIZE	NP	1
LOWER_BOUND	RP	1
UPPER_BOUND	RP	1
DATA	NP	[SIZE]*[SIZEOF(NP)]

Table A.19 -- PIKS objects data structures *(continued)*

PIKS OBJECT	Parameters	Length
OBJECT_CLASS “PIKS_LOOKUP_TABLE”	CS	Max 32
LUT_ID	CS	17
ENTRIES	NP	1
BANDS	NP	1
INPUT_DATATYPE	SP	1
OUTPUT_DATATYPE	SP	1
DATA	BD,ND,SD,RD,CD	[ENTRIES]*[BANDS]* [SIZEOF(OUTPUT_DATATYPE)]

Table A.19 -- PIKS objects data structures *(continued)*

PIKS OBJECT	**Parameters**	**Length**
OBJECT_CLASS “PIKS_MATRIX”	CS	Max 32
MATRIX_ID	CS	Max 32
COLUMNS	NP	1
ROWS	NP	1
DATATYPE	SP	1
DATA	ND,SD,RD,CD	[COLUMNS]*[ROWS]* [SIZEOF(DATATYPE)]

Table A.19 -- PIKS objects data structures *(continued)*

PIKS OBJECT	**Parameters**	**Length**
OBJECT_CLASS “PIKS_NEIGHBOURHOOD”	CS	18
ARRAY_ID	CS	Max 32
X_SIZE	NP	1
Y_SIZE	NP	1
Z_SIZE	NP	1
T_SIZE	NP	1
B_SIZE	NP	1
X_KEY_PIXEL	SP	1
Y_KEY_PIXEL	SP	1
Z_KEY_PIXEL	SP	1
T_KEY_PIXEL	SP	1
B_KEY_PIXEL	SP	1
SCALE_FACTOR	SP	1
LABEL	SP	1
DATA_TYPE	SP	1
DATA	BD,ND,SD,RD,CD	[X_SIZE]*[Y_SIZE]*[Z_SIZE]* [T_SIZE]*[B_SIZE]* [SIZEOF(DATATYPE)]

Table A.19 -- PIKS objects data structures *(continued)*

PIKS OBJECT	**Parameters**	**Length**
OBJECT_CLASS “PIKS_PIXEL_RECORD”	CS	17
RECORD_ID	CS	Max 32
B	NP	1
BAND_1_DATATYPE	SP	1
BAND_2_DATATYPE	SP	1
…	…	…
BAND_B_DATATYPE	SP	1
DATA	BD,ND,SD,RD,CD	[SIZEOF(BAND_1_DATATYPE)]* [SIZEOF(BAND_2_DATATYPE)]*…* [SIZEOF(BAND_B_DATATYPE)]

Table A.19 -- PIKS objects data structures *(continued)*

PIKS OBJECT	Parameters	Length
OBJECT_CLASS "PIKS_STATIC"	CS	11
STATIC_ID	CS	Max 32
X_SIZE	NP	1
Y_SIZE	NP	1
Z_SIZE	NP	1
T_SIZE	NP	1
B_SIZE	NP	1
LABEL	SP	1
DATATYPE	SP	1
DATA	BD,ND,SD,RD,CD	[X_SIZE]*[Y_SIZE]*[Z_SIZE]* [T_SIZE]*[B_SIZE]* [SIZEOF(DATATYPE)]

Table A.19 -- PIKS objects data structures *(continued)*

PIKS OBJECT	Parameters	Length
OBJECT_CLASS "PIKS_TUPLE"	CS	10
TUPLE_ID	CS	Max 32
ENTRIES	NP	1
DATATYPE	SP	1
DATA	BD,ND,SD,RD,CD,CS	[ENTRIES]*[SIZEOF(DATATYPE)]

Table A.19 -- PIKS objects data structures *(continued)*

PIKS OBJECT	Parameters	Length
OBJECT_CLASS "PIKS_ROI_ARRAY"	CS	14
ROI_ARRAY_ID	CS	Max 32
X_SIZE	NP	1
Y_SIZE	NP	1
Z_SIZE	NP	1
T_SIZE	NP	1
B_SIZE	NP	1
DATA	BP	[[X_SIZE]*[Y_SIZE]*[Z_SIZE]* [T_SIZE]*[B_SIZE]] / [8]

Table A.19 -- PIKS objects data structures *(continued)*

PIKS OBJECT	Parameters	Length
OBJECT_CLASS "PIKS_ROI_COORDINATE"	CS	19
ROI_COORDINATE_ID	CS	Max 32
X_SIZE	NP	1
Y_SIZE	NP	1
Z_SIZE	NP	1
T_SIZE	NP	1
B_SIZE	NP	1
COLLECTION_SIZE	NP	1
POLARITY	NP	1
DATA	NP	[SIZE]*[5]*[SIZEOF(NP)]

Table A.19 -- PIKS objects data structures *(continued)*

PIKS OBJECT	Parameters	Length
OBJECT_CLASS "PIKS_ROI_ELLIPTICAL"	CS	19
ROI_ELLIPTICAL_ID	CS	Max 32
X_SIZE	NP	1
Y_SIZE	NP	1
Z_SIZE	NP	1
T_SIZE	NP	1
B_SIZE	NP	1
X_ELLIPSE_CENTRE	NP	1
Y_ELLIPSE_CENTRE	NP	1
Z_ELLIPSE_CENTRE	NP	1
T_ELLIPSE_CENTRE	NP	1
B_ELLIPSE_CENTRE	NP	1
X_ELLIPSE_LENGTH	NP	1
Y_ELLIPSE_LENGTH	NP	1
Z_ELLIPSE_LENGTH	NP	1
T_ELLIPSE_LENGTH	NP	1
B_ELLIPSE_LENGTH	NP	1
X_INDEX_MANIPULATE	NP	1
Y_INDEX_MANIPULATE	NP	1
Z_INDEX_MANIPULATE	NP	1
T_INDEX_MANIPULATE	NP	1
B_INDEX_MANIPULATE	NP	1
DIMENSIONALITY	NP	1
POLARITY	NP	1

Table A.19 -- PIKS objects data structures *(continued)*

PIKS OBJECT	Parameters	Length
OBJECT_CLASS "PIKS_ROI_POLYGON"	CS	16
ROI_POLYGON_ID	CS	Max 32
X_SIZE	NP	1
Y_SIZE	NP	1
Z_SIZE	NP	1
T_SIZE	NP	1
B_SIZE	NP	1
ENTRIES	NP	1
X_INDEX_MANIPULATE	NP	1
Y_INDEX_MANIPULATE	NP	1
Z_INDEX_MANIPULATE	NP	1
T_INDEX_MANIPULATE	NP	1
B_INDEX_MANIPULATE	NP	1
POLARITY	NP	1
DATA	NP	[ENTRIES]*[2]*[SIZEOF(NP)]

Table A.19 -- PIKS objects data structures *(continued)*

PIKS OBJECT	Parameters	Length
OBJECT_CLASS "PIKS_ROI_RECTANGULAR"	CS	20
ROI_RECTANGULAR_ID	CS	Max 32
X_SIZE	NP	1
Y_SIZE	NP	1
Z_SIZE	NP	1
T_SIZE	NP	1
B_SIZE	NP	1
X_START	NP	1
Y_START	NP	1
Z_START	NP	1
T_START	NP	1
B_START	NP	1
X_END	NP	1
Y_END	NP	1
Z_END	NP	1
T_END	NP	1
B_END	NP	1
X_INDEX_MANIPULATE	NP	1
Y_INDEX_MANIPULATE	NP	1
Z_INDEX_MANIPULATE	NP	1
T_INDEX_MANIPULATE	NP	1
B_INDEX_MANIPULATE	NP	1
DIMENSIONALITY	NP	1
POLARITY	NP	1

Table A.19 -- PIKS objects data structures *(continued)*

PIKS OBJECT	Parameters	Length
OBJECT_CLASS "PIKS_VALUE_BOUNDS"	CS	17
BOUNDS_ID	CS	Max 32
SIZE	NP	1
LOWER_BOUND	SP	1
UPPER_BOUND	SP	1
DATATYPE	SP	1
DATA	NP,ND,SD,RD	[SIZE]*[[5]*[SIZEOF(NP)]+ [SIZEOF(DATATYPE)]]

Table A.19 -- PIKS objects data structures *(concluded)*

PIKS OBJECT	Parameters	Length
OBJECT_CLASS "PIKS_IMAGE"	CS	10
PIKS_IMAGE_ID	CS	Max 32
X_SIZE	NP	1
Y_SIZE	NP	1
Z_SIZE	NP	1
T_SIZE	NP	1
B_SIZE	NP	1
X_BANDTYPE	NP	1
Y_BANDTYPE	NP	1
Z_BANDTYPE	NP	1
T_BANDTYPE	NP	1
B_BANDTYPE	NP	1
X_WHITE_POINT	RP	1
Y_WHITE_POINT	RP	1
Z_WHITE_POINT	RP	1
STRUCTURE	SP	1
COLOUR_SPACE	SP	1
DATA Note - For this object the image data is located in the BIIF_ATTACHMENT_LEVEL associated with this TFS Profile.	BD,ND,SD,RD,CD	[X_SIZE]*[SIZEOF(X_BANDTYPE)]* [Y_SIZE]*[SIZEOF(Y_BANDDYPE)]* [Z_SIZE]*[SIZEOF(Z_BANDTYPE)]* [T_SIZE]*[SIZEOF(T_BANDTYPE)]* [B_SIZE]*[SIZEOF(B_BANDTYPE)]

A.2.6 TFS escape command

The TFS ESCAPE command conforms to the generic TFS command format with TS = 60, TSS = 10, and non-zero CPL. This is an application-specific command that is to be defined by an application Profile.

Annex B
(normative)

Vector Quantization

B.1 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 an BIIF VQ file is contained within the BIIF 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.

B.2 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 B.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 BIIF file, as illustrated in Figure B.2.

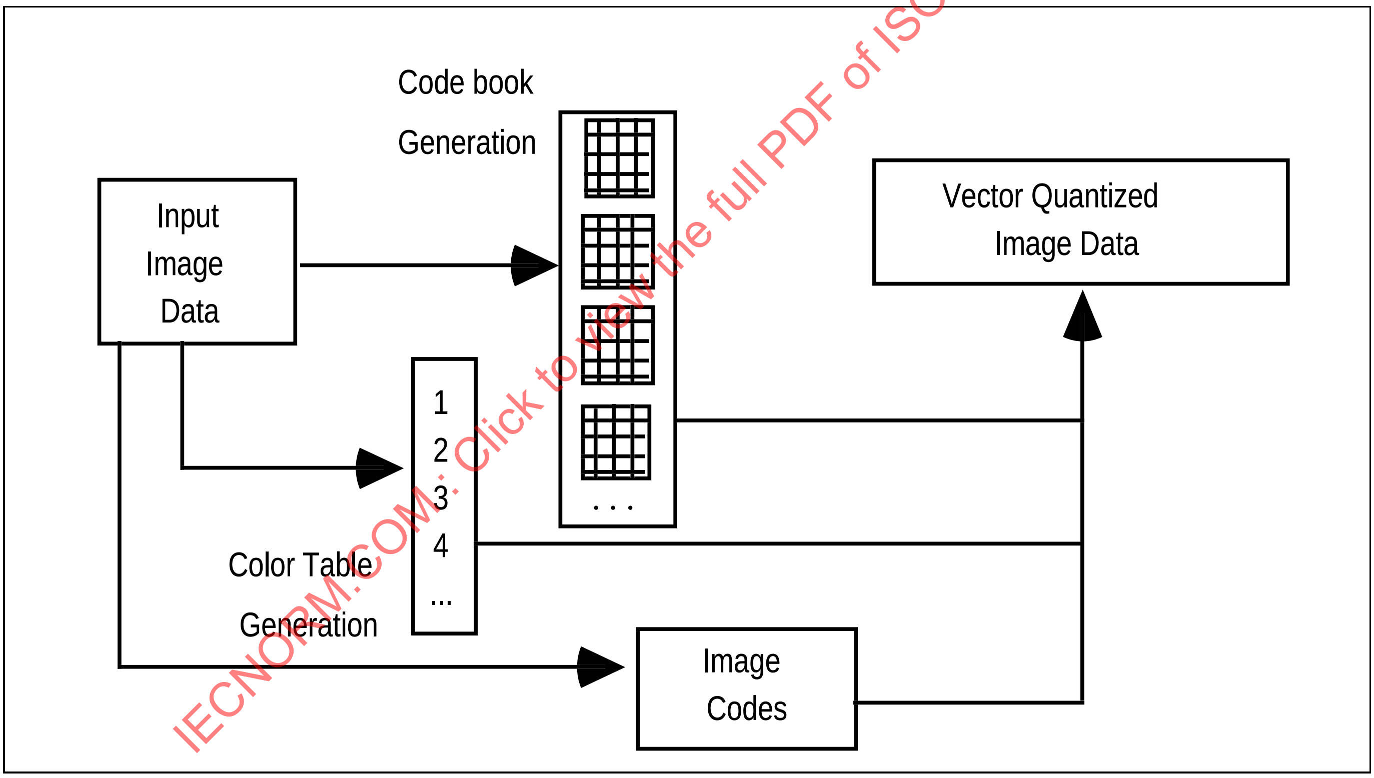


Figure B.1 -- Vector quantization process flow

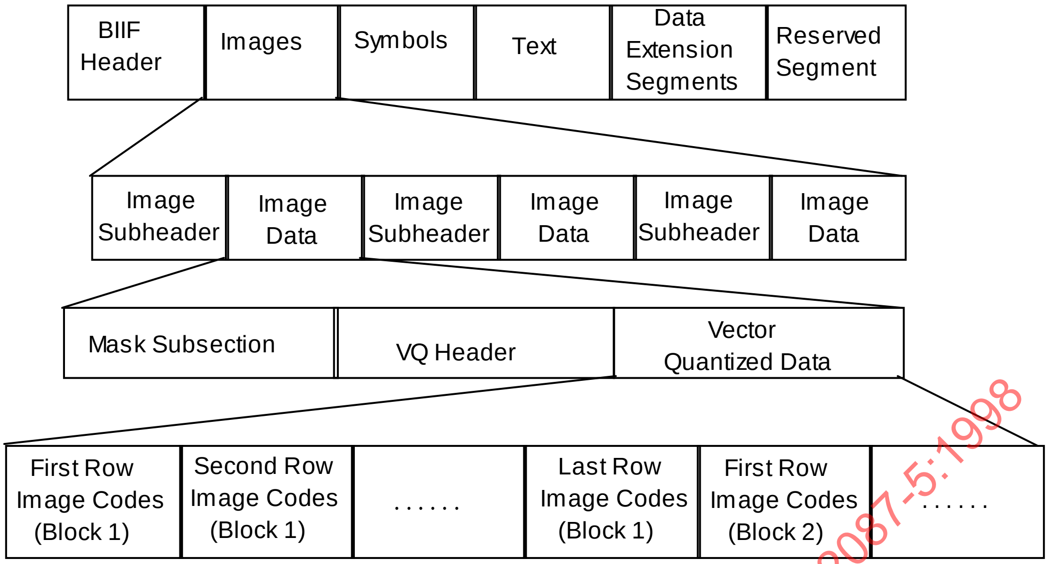


Figure B.2 -- BIIF file structure with VQ data

B.3 Reconstruction

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 within BIIF in terms of the types and sizes of color lookup tables allowed. However, current implementation of VQ within BIIF uses a single RGB/LUT. Other organizations may be implemented in the future.

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 B3. 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.

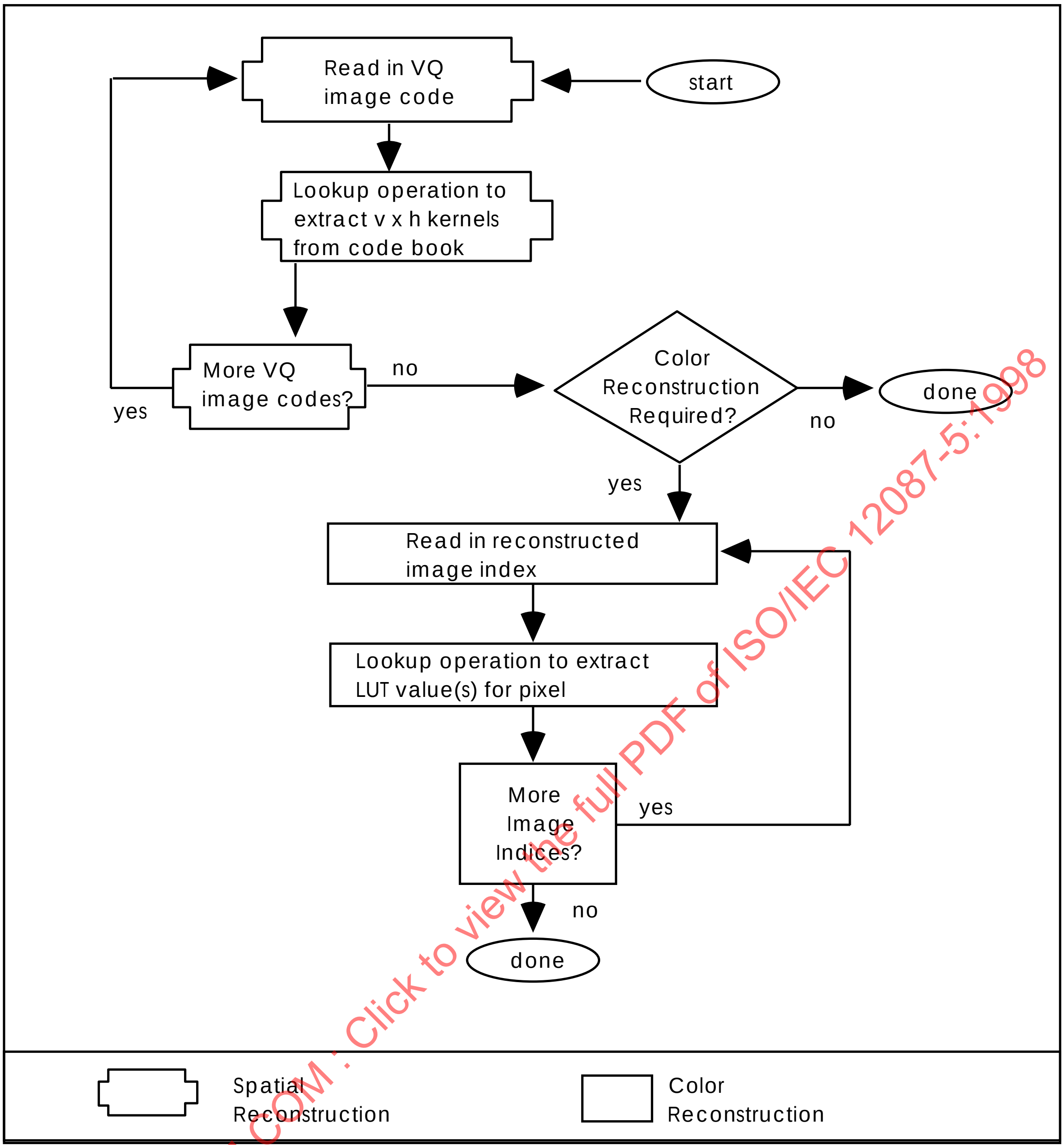


Figure B.3 -- VQ reconstruction procedure

B.3.1 Spatial reconstruction

When the BIIF IC field is set to C4 or M4, the image data field of the VQ formatted BIIF 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.

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 BIIF structure allows for the organization of the VQ code book to be optimized for the specific use of the VQ data. While some BIIF 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.

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 B4 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.

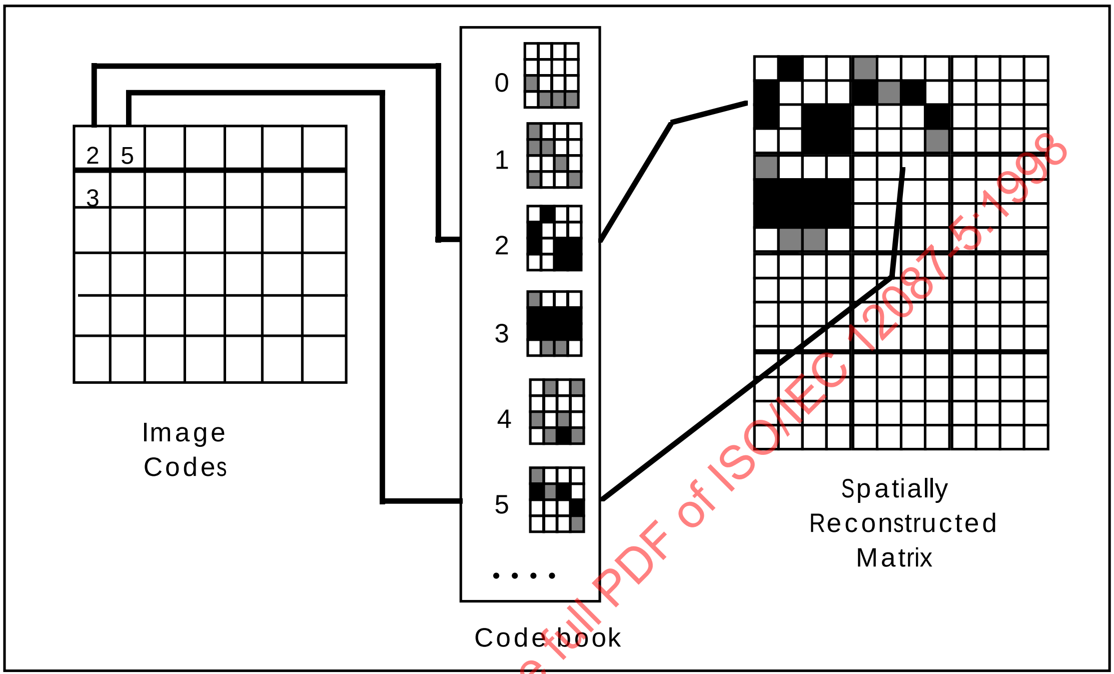


Figure B.4 — Spatial reconstruction

B.3.2 Color reconstruction

Current implementation of VQ within BIIF 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 in the BIIF image subheader 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 B.5

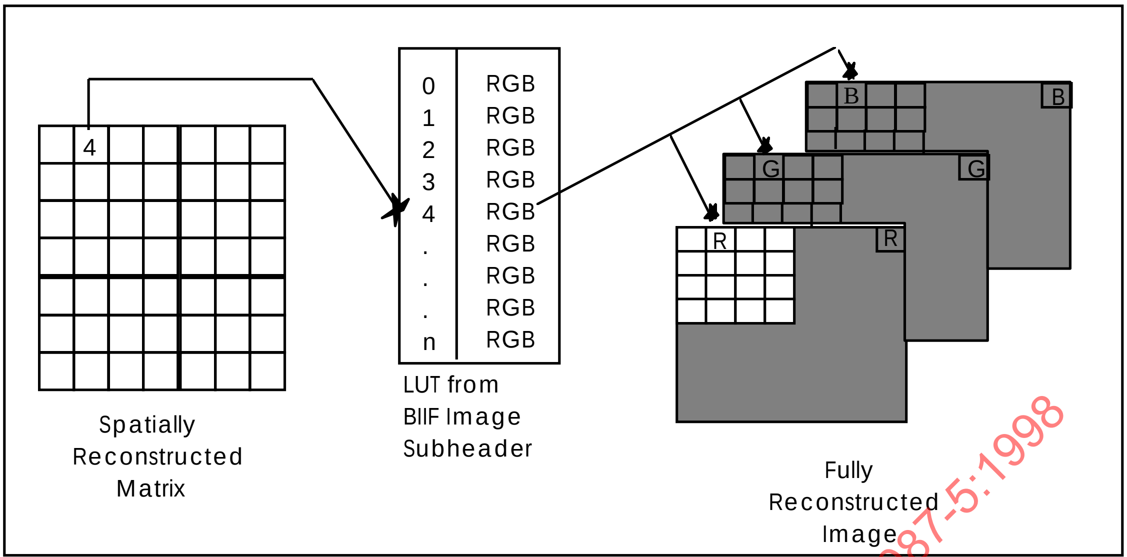


Figure B.5 -- Color reconstruction

B.3.3 Data elements

The BIIF VQ format allows for many quantization ratios recorded in the COMRAT fields, and several organizations of the code books and color tables. The BIIF 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 BIIF 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 BIIF format structure, vector quantized data elements sometimes use compression/decompression terms in effort to maintain configuration within the BIIF.

B.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 BIIF 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 BIIF 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.

B.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 BIIF standard.

B.3.3.3 Code book organization

The BIIF 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 (BIIF image subheader field NPPBV) and number of pixels per block horizontal (BIIF 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.

B.3.3.4 Spatial data section

The spatial data section of the BIIF 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 BIIF 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 BIIF 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 BIIF image subheader and the number of blocks per column (NBPC) field of the BIIF 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].

B.4 File organization

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

```
{1}
[BIIF 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 B.6 -- Structure of the BIIF VQ image data section

B.5 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 [BIIF image data] 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 ³ 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].

B.6 Definitions - BIIF header and image subheader

Table B.1 provides specific data values for BIIF header and image subheader fields particular to VQ data.

Table B.1 -- BIIF 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 BIIF 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 C
(normative)

Profiling BIIF

C.1 Profiling process

This Annex provide tables that are to be filled in to define a selective application of BIIF. Completion of all the tables below and supporting documents for special definitions constitute a "profile" of BIIF. (In ISO functional profiling taxonomy, this type of profile is referred to as a "format profile" since it defines a data format for use in interchange or sharing of information.)

Profiling begins by identifying the BIIF file parts that are required to support the application and the fields within each file part that are affected. Tables C.1, C.3, C.4, C.5, C.6 and C.7 correspond to Tables 1, 3, 4, 6, 7 and 8 of Subclause 4 respectively. Table C.8a and C.8b correspond to tables 9, 10, 11 and 12 of Subclause 4.

C.2 Profile proforma

C.2.1 Use of the model profile

The BIIF Model Profile is provided as an exemplary conformant usage of BIIF. As indicated in clauses below, the profile designer may elect to accept the Model Profile value for each part and field simply by indicating "Same as Model Profile" in the appropriate column.

A Profile Variant (PV) set for each file part is included in the tables of this clause. PVV fields have a complete syntactic and semantic definition in this standard (in clause 4), but typically may have further restriction of value ranges for a particular application. The Model Profile also addresses use of PVU fields by defining structure and representative values for the fields. As such, the Model Profile provides a recommended starting point for other profiles.

C.2.2 Rules for filling out the proforma tables

All tables below are to be completed in accordance with the following rules and recommendations. Developers are encouraged to accept the Model Profile definitions for all PVU fields, and given the opportunity, where appropriate, to accept the Model Profile constraints for PVV fields. The rules below (designated R1 through R10) shall be followed in filling out the proforma tables for any profile that is to be nominated for registration and/or progression as an ISP.

R1 An expression of value range or a list of allowable BCS entries and a definition of the significance of each entry. When no list or range is present, the default will be to fill the field with spaces (0x20) for BCS-A fields and zeros (0x30) for BCS-N fields.

R2 The profile definition may constrain allowable field entries to further define the allowable syntax in a manner similar to that of the date and time fields, e.g. CCYYMMDDhhmmss.

R3 The profile definition may define sub-fields. The definition shall be done in a similar manner as to that used in Clause 4 for defining fields in this standard. A separately documented definition may be referenced in the proforma and attached to the proforma.

R4 The profile definition for security-related fields shall be identical for all header and subheaders (i.e., FSEC, ISCSEC, SSSEC, TSSEC, etc.). A separately documented definition may be referenced in the proforma and attached to the proforma.

R5 The profile may indicate a numeric value range for the field in the form of Minimum-value - Maximum-value (inclusive) specification. The values may be integers, or in some cases other numeric types as applicable to the fields defined in Clause 4.

R6 Where indicated, a unique string identifier shall be provided. The domain across which uniqueness must be maintained is further clarified as follows: 1) within a BIIF file instance, 2) within profile, 3) within ISO standards, and within an authorized registry.

R7 For some fields there must be a unique NULL/ZERO/NONE indication because of a conditional field dependency on this value. This indicator is usually "N" and shall appear first in the enumerated set of values, for example: {"N", "KEY1", "KEY2"}

R8 The fields marked U8 (UTF-8) are designated to contain a textual comment (general information, name, etc.) that may be language specific but is otherwise unstructured.

R9 For PVV designated fields, the profile definer may indicate the value constraint on the field as follows:

a) Any value permitted by the standard (as specified in Clause 4);
b) A specific set of values or single value from an enumerated set (Rule 1) or a range of values (Rule 5) as permitted by the standard; or,
c) The Model Profile constraint (by indicating “Same as Model Profile”).

R10 For PVU designated fields, the profile definer is required to provide a complete discussion of the field use, including the following points:

a) Any substructuring of the field (Rule 3);
b) Any syntactic constraint against either the entire field and/or sub-fields (Rule 2); and,
c) An explanation of the meaning of the field and its subfields, including detailed examples of usage.

C.2.2.1 Profile Tables

The following clauses provide the proforma (tables) for profile specification. They also specify the model profile of BIIF.

C.2.2.1.1 BIIF File Header

Table C.1 is the proforma for profile specification of BIIF file header fields.

Table C.1 -- File header fields

FIELD	TYPE	CE / SIZE	PROFILE OPTIONS & RULES	MODEL PROFILE	NEW PROFILE SPECIFICATION Profile Name:__________
	N/A	N/A	Specify whether this new profile specification includes the entire model profile BFMP as a capability in addition to the new profile capabilities (Clause 5.1).		__ Model profile fully included (can produce and/or read both BFMP files and the new profile specification files). __ Model profile not fully included (can only produce and/or read the new profile specification files).
FHDR	R/PVV	A/4	Unique profile name not already registered.	BFMP	__ __ __ __
FVER	R/PVV	A/5	Version identifier unique to the registered profile name.	01.00	__ __ __ __ __
CLEVEL	R/PVV	A/2	A value of 00 indicates the profile has no internal hierarchy. If an internal hierarchy is desired, the profile may designate two or more values in the range 01-99 to differentiate parameters, values, and ranges within the internal hierarchy. Each profile entry shall specify the CLEVEL hierarchy constraints where applicable. When profile entries do not specify CLEVEL constraints, they are applicable across all CLEVELs. The profile entry shall also define the extent of the Common Coordinate System (CCS) applicable for each designated CLEVEL.	01 02 The CCS extent for each CLEVEL is: 01 2048 x 2048 02 8192 x 8192	Select ONE of the following: ___ Same as Model Profile ___ CLEVEL=00 The CCS extent=________ ___ List allowed values and CCS extent below. Enter CLEVEL specification for each applicable field in the profile. List:
STYPE	R	A/4	Always BF01 for all profiles of this standard.	BF01	_X_ Same as Model Profile

Table C.1 -- File header fields (continued)

FIELD	TYPE	CE / SIZE	PROFILE OPTIONS & RULES	MODEL PROFILE	NEW PROFILE SPECIFICATION Profile Name:____________
OSTAID	R/PVU	A/10	Any BCS-A string is allowed. Profile may specify further constraints.	Any BCS-A string.	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
FDT	R	N/14	As specified in Table 1 for all profiles.	CCYYMMDDhhmmss	_X_ Same as Model Profile
FTITLE	R/PVU	U8/80	Any UTF-8 string is allowed. Profile may specify further constraints.	Any BCS-A string.	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
FSEC	R/PVU	U8/167	The profile definition for security-related fields shall be identical for all header and subheaders (i.e., FSEC, ISCSEC, SSSEC, TSSEC, etc.). The profile definition may define subfields and associated constraints using Table C.2.	See Table C.2	Select ONE of the following: ___ Same as Model Profile ___ The profile specific constraints are as specified in the new profile specification of Table C.2.
FSCOP	R/PVV	N/5	Any BCS-N string is allowed. Profile may specify further constraints.	00000-99999 An entry of 00000 shall mean that no count of copies is being maintained.	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
FSCPYS	R/PVV	N/5	Any BCS-N string is allowed. Profile may specify further constraints.	00000-99999 An entry of 00000 shall mean that no total file count is being tracked.	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
ENCRYP	R/PVU	A/1	Any BCS-A character is allowed. 0=not encrypted other codes defined by profile	0=not encrypted	Select ONE of the following: ___Same as Model Profile ___The following are profile specified constraints:

Table C.1 -- File header fields *(continued)*

FIELD	TYPE	CE / SIZE	PROFILE OPTIONS & RULES	MODEL PROFILE	NEW PROFILE SPECIFICATION Profile Name:___________
OID	R/PVU	U8/45	Any UTF-8 string is allowed. Profile may define subfields and specify associated constraints.	Any BCS-A string. No subfields defined.	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
FL	R/PVV	N/12	As specified in Table 1 for all profiles. Overall file size constraints may be imposed by profile.	Not to exceed 2 Gigabytes -1, (2147483647).	Select ONE of the following: ___Same as Model Profile ___The following are profile specified constraints:
HL	R	N/6	As specified in Table 1 for all profiles.	Calculated value.	_X_ Same as Model Profile
NUMI	R/PVV	N/3	Profile shall specify the number or range of image segments allowed in a BIIF file conforming to the profile.	CLEVEL 01: 000-001 CLEVEL 02: 000-020	Select ONE of the following: ___ Same as Model Profile ___ The profile specified range is:
LISHn & LIn	C C	N/6 N/10	A pair of length values as specified in Table 1. These fields repeat (in pairs) the number of times identified by NUMI.	Calculated.	_X_ Same as Model Profile
NUMS	R	N/3	Profile shall specify the number or range of symbol segments allowed in a BIIF file conforming to the profile.	CLEVEL 01: 000 CLEVEL 02: 000-100	Select ONE of the following: ___ Same as Model Profile ___ The profile specified range is:
LSSHn & LSn	C C	N/4 N/6	A pair of length values as specified in Table 1. The fields repeat (in pairs) the number of times identified by NUMS.	Calculated.	_X_ Same as Model Profile
NUMX	R	N/3	Always 000 for all profiles.	000	_X_ Same as Model Profile
NUMT	R	N/3	Profile shall specify the number or range of text segments allowed in a BIIF file conforming to the profile.	CLEVEL 01: 000 CLEVEL 02: 000-010	Select ONE of the following: ___ Same as Model Profile ___ The profile specified range is:
LTSHn & LTn	C C	N/4 N/5	A pair of length values as specified in Table 1. The fields repeat (in pairs) the number of times identified by NUMT.	Calculated.	_X_ Same as Model Profile

Table C.1 -- File header fields *(continued)*

FIELD	TYPE	CE / SIZE	PROFILE OPTIONS & RULES	MODEL PROFILE	NEW PROFILE SPECIFICATION Profile Name:__________
NUMDES	R	N/3	Profile shall specify the number or range of data extension segments allowed in a BIIF file conforming to the profile. When DESs are allowed, the profile shall identify the specific DESs supported at the time of profile registration. Additional DES types can be added to the profile through the registration process.	CLEVEL 01: 000 CLEVEL 02: 000-020 Supported DESs: DESTAG: TRE_OVERFLOW DESTAG: TRANSPORTABL E_FILE_STRUCT	Select ONE of the following: ___ Same as Model Profile ___ The profile specified range & list is: Supported DESs:
LDSHn & LDn	C C	N/4 N/9	A pair of length values as specified in Table 1. The fields repeat in pairs the number of times identified by NUMDES.	Calculated.	_X_ Same as Model Profile
NUMRES	R	N/3	Profile shall specify the number or range of registered extension segments allowed in a BIIF file conforming to the profile. When RESs are allowed, the profile shall identify the specific DESs supported at the time of profile registration. Additional RES types may be added to the profile through the registration process.	CLEVEL 01: 000 CLEVEL 02: 000 Supported RESs: None	Select ONE of the following: ___ Same as Model Profile ___ The profile specified range is: Supported RESs:
LRSHn & LRn	C C	N/4 N/7	A pair of length values as specified in Table 1. The fields repeat in pairs the number of times identified by NUMRES.	Calculated.	_X_ Same as Model Profile
UDHDL	R	N/5	As specified in Table 1 for all profiles.	Calculated.	_X_ Same as Model Profile
UDHOFL	C	N/3	As specified in Table 1 for all profiles.	Calculated.	_X_ Same as Model Profile

Table C.1 -- File header fields *(concluded)*

FIELD	TYPE	CE / SIZE	PROFILE OPTIONS & RULES	MODEL PROFILE	NEW PROFILE SPECIFICATION Profile Name:___________
UDHD	C/PVU		As specified in Table 1 for all profiles.	Any tagged record extension general to the BIIF file.	__ Same as Model Profile Select all that apply: ___ TREs are prohibited ___ Any Public TRE ___ Any Private TRE ___ Allowed Public TREs List: ___ Allowed Private TREs List: Specify constraints:
XHDL	R/PVV	N/5	As specified in Table 1 for all profiles.	Calculated.	_X_ Same as Model Profile
XHDLOFL	C	N/3	As specified in Table 1 for all profiles.	Calculated.	_X_ Same as Model Profile
XHD	C/PVU		As specified in Table 1 for all profiles.	Any tagged record extension general to the BIIF file.	__ Same as Model Profile Select all that apply: ___ TREs are prohibited ___ Any Public TRE ___ Any Private TRE ___ Allowed Public TREs List: ___ Allowed Private TREs List: Specify constraints:

C.2.2.1.2 BIIF security fields

Table C.2 is the proforma for profile specification of BIIF security fields.

Table C.2 -- Security fields specification

FIELD	TYPE	CE/ SIZE	PROFILE OPTIONS & RULES	MODEL PROFILE	NEW PROFILE SPECIFICATION Profile Name:____________
FSEC ISCSEC SSSEC TSSEC DESCLAS RESCLAS	R R R R R R	U8/167 U8/167 U8/167 U8/167 U8/167 U8/167	The profile definition for security-related fields shall be identical for all header and subheaders (FSEC, ISCSEC, SSSEC, TSSEC, DESCLAS, RESCLAS). The profile definition may define subfields and associated constraints using this table.	The model profile specification for subfields is as described below.	Select ONE of the following: ___ Same as Model Profile ___ The profile specific definition and constraint are specified below.
MODEL PROFILE SPECIFICATION					
SUBFIELD	**TYPE**	**CE/ SIZE**	**DESCRIPTION**	**VALUE RANGE**	**NOTES**
SEC1	R	A/1	Security code applicable to the entire file. The only defined code for the model profile is: U - Unspecified.	U	The model profile does not detail specific file security parameters.
SEC2	R	U8/166	Free form text. Any set of UTF-8 characters is allowed. Default entry is all BCS-A spaces.	Any BCS-A. Default is all BCS-A spaces.	
NEW PROFILE SPECIFICATION					
SUBFIELD	**TYPE**	**CE/ SIZE**	**DESCRIPTION**	**VALUE RANGE**	**NOTES**

C.2.2.1.3 BIIF image subheader fields

Table C.3 is the proforma for profile specification of BIIF image subheader fields.

Table C.3 -- Image subheader fields

FIELD	TYPE	CE/ SIZE	PROFILE OPTIONS & RULES	MODEL PROFILE	NEW PROFILE SPECIFICATION Profile Name:____________
IM	R	A/2	Always IM for all profiles.	IM	_X Same as Model Profile
IID	R/PVU	A/10	Any BCS-A string is allowed. Profile may specify further constraints.	Any BCS-A string.	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
IDATIM	R	N/14	As specified in Table 3 for all profiles.	CCYYMMDDhhmmss	_X Same as Model Profile
IINFO	R/PVU	U8/97	Any UTF-8 string is allowed. The profile definition may define subfields and associated constraints.	Any BCS-A string.	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
ISCSEC	R/PVU	U8/167	As specified in Table C.2.	As specified in Table C.2.	As specified in Table C.2
ENCRYP	R/PVU	A/1	Any BCS-A character code is allowed. Profile shall define the meaning of each code.	0 = not encrypted.	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
ISORCE	R/PVU	U8/42	Any UTF-8 string is allowed. The profile definition may define subfields and associated constraints.	Any BCS-A string.	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
NROWS	R/PVV	N/8	Profile shall specify the range for the number of rows of pixel values allowed in a BIIF image conforming to the profile.	CLEVEL 01: 00000001-00002048 CLEVEL 02: 00000001-00008192	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:

Table C.3 -- Image subheader fields *(continued)*

FIELD	TYPE	CE/ SIZE	PROFILE OPTIONS & RULES	MODEL PROFILE	NEW PROFILE SPECIFICATION Profile Name:___________
NCOLS	R/PVV	N/8	Profile shall specify the range for the number of columns of pixel values allowed in a BIIF file conforming to the profile.	CLEVEL 01: 00000001-00002048 CLEVEL 02: 00000001-00008192	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
PVTYPE	R/PVV	A/3	Select from the following allowed values: B INT SI R C	CLEVEL 01: INT CLEVEL 02: B INT	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
IREP	R/PVV	A/8	Select from the following values: MONO RGB RGB/LUT HIS CMY CMYK YIQ YUV YCbCr CIE 1D 2D ND MULTI PIKS1, PIKS2, PIKS3, PIKS4 ----and/or---- Specify additional values and their specified use.	CLEVEL 01: MONO CLEVEL 02: MONO RGB RGB/LUT YCbCr PIKS1	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:

Table C.3 -- Image subheader fields *(continued)*

FIELD	TYPE	CE / SIZE	PROFILE OPTIONS & RULES	MODEL PROFILE	NEW PROFILE SPECIFICATION Profile Name:____________
ICAT	R/PVU	A/8	Select from the following values: VIS - Visual imagery SL - Side looking radar TI - Thermal infrared FL - Forward looking IR RD - Radar EO - Electro-optical OP - Optical HR - High res. radar HS - Hyperspectral CP - Color frame photo BP - Black/white photo SAR - Synthetic Aperture Radar SARIQ - SAR radio hologram IR - Infrared MS - Multi-spectral FP - Finger prints MRI - Magnetic Resonance imagery XRAY - x-rays CAT - CAT scan MAP - Raster map PAT - Colour patch LEG - Legend DTEM - Elevation model data MATR - general matrix data LOCG - Location grids ----and/or---- Specify additional values and their specified use.	CLEVEL 01: VIS CLEVEL 02: Select from the following list: VIS EO SAR IR MRI XRAY CAT MAP PAT LEG	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
ABPP	R/PVV	N/2	Select from the following options: 01, 04, 08, 09, 10, 11, 12, 16, 32, 64, 01-08, 09-16, 17-32, 33-64, 65-96 ----and/or ---- Specify additional values and their specified use.	CLEVEL 01: 08 CLEVEL 02: 01, 08, 09, 10, 11, 12	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:

Table C.3 -- Image subheader fields *(continued)*

FIELD	TYPE	CE / SIZE	PROFILE OPTIONS & RULES	MODEL PROFILE	NEW PROFILE SPECIFICATION Profile Name:___________
PJUST	R/PVV	A/1	Select from the following options: R L	R	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
ICORDS	R/PVV	A/1	Space character - or - Specify and define codes.	CLEVEL 01: Space CLEVEL 02: Space	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
IGEOLO	C/PVU	A/60	For each code specified in ICORDS, define the content and structure of this field.	Omitted.	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
NICOM	R/PVV	N/1	Select any range within 0-9.	0-9	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
ICOMn	C/PVU	U8/80	Any UTF-8 string is allowed. Profile may specify further constraints.	Any BCS-A string.	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:

Table C.3 -- Image subheader fields *(continued)*

FIELD	TYPE	CE / SIZE	PROFILE OPTIONS & RULES	MODEL PROFILE	NEW PROFILE SPECIFICATION Profile Name:___________
IC	R/PVV	A/2	Any BCS-A string is allowed to identify the type of compression used in the image data field. The character M is reserved for use to indicate that image data mask tables have been included in the image data field. Reference must be made to the specification document describing the compression. Representative values are: NC - uncompressed NM - uncompressed with mask tables	CLEVEL 01: NC - Uncompressed CLEVEL 02: NC- Uncompressed NM - Uncompressed with mask table(s). C1/M1 - Bi-Tonal compression per ITU-T T.4, AMD2 08/95 C3/M3 - JPEG lossy DCT compression per ISO/IEC 10918-1 and ISO/IEC 10918-3. C4/M4 - Vector Quantization as outlined in Appendix B C5/M5 - JPEG lossless compression per ISO/IEC 10918-1 and ISO/IEC 10918-3.	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:

Table C.3 -- Image subheader fields *(continued)*

FIELD	TYPE	CE / SIZE	PROFILE OPTIONS & RULES	MODEL PROFILE	NEW PROFILE SPECIFICATION Profile Name:____________
COMRAT	C/PVU	A/4	The field is omitted when IC=NC or IC=NM. When present, the profile shall describe the use of the field for describing compression rate related information.	This conditional field is only present when the IC code is other than NC or NM. ELSE: For C1/M1: 1D - one dimensional coding 2DS- two dimensional coding, standard vertical resolution (K=2) 2DH- two dimensional coding, high vertical resolution (K=4) See ITU-T T.4, AMD2 08/95. For C3/M3 the value is always 00.0. For C4/M4 the value is in the form of n.nn representing the approximate number of bits per pixel for the compressed image. For C5/M5 the value is always 0.00.	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
NBANDS	R/PVV	A/1	Identify allowed values from: 0 1-9 T	CLEVEL 01: 1 CLEVEL 02: 1 3	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
XBANDS	C/PVV	N/5	Conditional field; omitted unless NBANDS=0.	CLEVEL 01: Not Used. CLEVEL 02: Not Used.	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
NOTE: The fields IREPBANDn through LUTnm repeat the number of times indicated in the NBANDS field or the XBANDS field.					

Table C.3 -- Image subheader fields *(continued)*

FIELD	TYPE	CE / SIZE	PROFILE OPTIONS & RULES	MODEL PROFILE	NEW PROFILE SPECIFICATION Profile Name:___________
IREPBANDn	R/PVV	A/2	Identify allowed value(s) for each band which further identifies the significance of the band as related to the value in the IREP field.	For IREP=MONO, two spaces. For IREP=RGB, R, G, B. For IREP=RGB/LUT two spaces. For IREP=YCbCr Y, Cb, Cr. For IREP=PIKS two spaces.	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
ISUBCATn	R/PVU	A/6	Identify allowed value(s) for each band which further identifies the significance of the band as related to the value in the ICAT field.	For all values of ICAT, the value is six spaces.	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
IFCn	C/PVU	A/1	Identify filter condition codes and the specification of the corresponding filter condition. The code N means there is no filter condition.	N	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
IMFLTn	C/PVU	A/3	Identify filter codes corresponding to each filter condition identified in IFCnn. If none, value is three spaces.	Three spaces.	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:

Table C.3 -- Image subheader fields *(continued)*

FIELD	TYPE	CE / SIZE	PROFILE OPTIONS & RULES	MODEL PROFILE	NEW PROFILE SPECIFICATION Profile Name:__________
NLUTSn	R/PVV	N/1	Identify the allowed range for specifying the number of look up tables (8-bit entries in each table) associated with the band. Typical values are: 0 - no LUTS. 1 - for translating NELUTS number of values to alternate 8-bit (or less) values. 2 - for translating NELUTS number of values into alternate values of 16 or less bits. 3 - For translating NELUTS number of values into alternate values of 24 or less bits.	CLEVEL 01: 0 CLEVEL 02: 0-3	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
NELUTn	C	N/5	When NLUTnn=00000, this field is omitted. Otherwise it specifies the number of 8-bit entries in each sequential LUT.	CLEVEL 01: Omitted. CLEVEL 02: Omitted when NLUTSn=0; otherwise 00001-32768	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
LUTDn1	C/PVU	Derived from Value NELUT1	This field shall be omitted if the n^{th} Band Number of LUTS is zero. Otherwise, this field shall contain the data defining the first look-up table for the nth image band. Multiple LUTs may be used to translate the index value into multiple octet values.	Data only	Same as Model Profile.
LUTDnm	C	Derived from Value NELUT1	This field shall be omitted if the n^{th} Band Number of LUTs is zero. Otherwise, this field shall contain the data defining the m^{th} look-up table for the n^{th} image band. Each entry in the look-up table is composed of one octet, ordered from most significant bit to least significant bit representing a value from 0 to 255.	Data only	Same as Model Profile

Table C.3 -- Image subheader fields *(continued)*

FIELD	TYPE	CE / SIZE	PROFILE OPTIONS & RULES	MODEL PROFILE	NEW PROFILE SPECIFICATION Profile Name:___________
ISYNC	R/PVV	A/1	Identify indicator codes for each allowed end of row or end of column marker to be used. For each listed code, identify the specification for the sync code marker. The value 0 indicates no sync code is used.	0	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
IMODE	R/PVV	A/1	IMODE codes are: B=Block interleaved R=Row interleaved P=Pixel Interleaved S=Band sequential	CLEVEL 01: B CLEVEL 02: B P R S	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints
NBPR	R/PVV	N/4	Identify the allowed range.	CLEVEL 01: 0001 CLEVEL 02: 0001 - 0064	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
NBPC	R/PVV	N/4	Identify the allowed range.	CLEVEL 01: 0001 CLEVEL 02: 0001 - 0064	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
NPPBH	R/PVV	N/4	Identify the allowed range.	CLEVEL 01: 0001 - 2048 CLEVEL 02: 0001 - 8192	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
NPPBV	R/PVV	N/4	Identify the allowed range.	CLEVEL 01: 0001 - 2048 CLEVEL 02: 0001 - 8192	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:

Table C.3 -- Image subheader fields *(continued)*

FIELD	TYPE	CE / SIZE	PROFILE OPTIONS & RULES	MODEL PROFILE	NEW PROFILE SPECIFICATION Profile Name:___________
NBPP	R	N/2	Select the allowed values from the following: 01 04 08 12 16 24 32 40 48 56 64 72 80 88 96 -- and/or -- specify additional values and specify use.	CLEVEL 01: 08 CLEVEL 02: 01 08 12 16	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
IDLVL	R	N/3	As specified in Table 3.	001-999	X Same as Model Profile
IALVL	R	N/3	As specified in Table 3.	000-998	X Same as Model Profile
ILOC	R	N/10	As specified in Table 3.	As specified in Table 3.	X Same as Model Profile
IMAG	R	A/4	As specified in Table 3.	As specified in Table 3.	X Same as Model Profile
UDIDL	R/PVV	N/5	As specified in Table 3 for all profiles.	Calculated.	_X_ Same as Model Profile
UDOFL	C	N/3	As specified in Table 3 for all profiles.	Calculated.	_X_ Same as Model Profile
UDID	C		As specified in Table 3 for all profiles.	Any tagged record extension specific to the image.	__ Same as Model Profile Select all that apply: ___ TREs are prohibited ___ Any Public TRE ___ Any Private TRE ___ Allowed Public TREs List: ___ Allowed Private TREs List: Specify constraints:
IXSHDL	R/PVV	N/5	As specified in Table 3 for all profiles.	Calculated.	_X_ Same as Model Profile
IXSOFL	C	N/3	As specified in Table 3 for all profiles.	Calculated.	_X_ Same as Model Profile

Table C.3 -- Image subheader fields *(concluded)*

FIELD	TYPE	CE / SIZE	PROFILE OPTIONS & RULES	MODEL PROFILE	NEW PROFILE SPECIFICATION Profile Name:____________
IXSHD	C/PVU		As specified in Table 3 for all profiles.	Any tagged record extension specific to the image.	__ Same as Model Profile Select all that apply: ___ TREs are prohibited ___ Any Public TRE ___ Any Private TRE ___ Allowed Public TREs List: ___ Allowed Private TREs List: Specify constraints:

C.2.2.1.4 BIIF image data mask table

Table C.4 is the proforma for profile specification of BIIF image data mask table fields.

Table C.4 -- Image data mask table

FIELD	TYPE	OCTETS	PROFILE OPTIONS & RULES	MODEL PROFILE	NEW PROFILE SPECIFICATION Profile Name:____________
When the profile specifies use of the image data mask table, the following fields all become applicable. See Table C.3, field IC. When the image data mask table is not specified in the profile, none of the following fields apply.			When selected codes for the IC field include the character M, the specification of image data mask tables becomes required.	CLEVEL 01: Image data mask tables not used CLEVEL 02: Used as specified in Table 4.	Select ONE of the following: ___ Same as Model Profile ___ NO. Image data mask tables are not allowed for use within this profile.
IMDATOFF	C	4	As specified in Table 4.	As specified in Table 4.	X Same as Model Profile
BMRLNTH	C	2	As specified in Table 4.	As specified in Table 4.	X Same as Model Profile
TMRLNTH	C	2	As specified in Table 4.	As specified in Table 4.	X Same as Model Profile
TPXCDLNTH	C	2	As specified in Table 4.	As specified in Table 4.	X Same as Model Profile
TPXCD	C	[1]Note	As specified in Table 4.	As specified in Table 4.	X Same as Model Profile
BMRnBNDm	C	4	As specified in Table 4.	As specified in Table 4.	X Same as Model Profile
TMRnBNDm	C	4	As specified in Table 4.	As specified in Table 4.	X Same as Model Profile
[1] The length of the TPXCD field is the next highest number of octets which can contain the number of bits identified in the TPXCDLNTH field (1 or 2 Octets) also as specified in Table 4.					

C.2.2.1.5 BIIF symbol subheader

Table C.5 is the proforma for profile specification of BIIF symbol subheader fields.

Table C.5 -- Symbol subheader

FIELD	TYPE	CE/ SIZE	PROFILE OPTIONS & RULES	MODEL PROFILE	NEW PROFILE SPECIFICATION Profile Name:___________
SY	R	A/2	Always SY for all profiles.	SY	_X Same as Model Profile
SID	R/PVU	A/10	Any BCS-A string is allowed. Profile may specify further constraints.	Any BCS-A string.	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
SNAME	R/PVU	U8/20	Any UTF-8 string is allowed. Profile may specify further constraints.	Any BCS-A string.	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
SSSEC	R/PVU	U8/167	As specified in Table C.2.	As specified in Table C.2.	As specified in Table C.2

Table C.5 -- Symbol subheader *(continued)*

FIELD	**TYPE**	**CE/ SIZE**	**PROFILE OPTIONS & RULES**	**MODEL PROFILE**	**NEW PROFILE SPECIFICATION Profile Name:____________**
ENCRYP	R/PVU	A/1	Any BCS-A character code is allowed. Profile shall define the meaning of each code.	0 = not encrypted.	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints.
SFMT	R/PVV	A/1	This field contains a C indicating the symbol data field contains data structured according to ISO/IEC 8632, Computer Graphics Metafile (CGM). Additional symbol format codes may be added through the graphical item registration process.	C	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
SSTRUCT	R/PVU	A/13	Any BCS-A string is allowed. The profile definition may define subfields and associated constraints.	0000000000000	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
SDLVL	R	N/3	001-999	As specified in Table 6.	_X Same as Model Profile
SALVL	R	N/3	000-998	As specified in Table 6.	_X Same as Model Profile
SLOC	R	N/10	As specified in Table 6.	As specified in Table 6.	_X Same as Model Profile
SLOC2	R/PVU	N/10	As specified in Table 6. Default is all zeros.	0000000000	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:

Table C.5 -- Symbol subheader *(concluded)*

FIELD	TYPE	CE/ SIZE	PROFILE OPTIONS & RULES	MODEL PROFILE	NEW PROFILE SPECIFICATION Profile Name:___________
SPARMS	R/PVU	A/13	Any BCS-A string is allowed. The profile definition may define subfields and associated constraints.	The 13 bytes are structured into two subfields as follows: **Subfield:** SCOLOR **Name:** Symbol Colour. This subfield contains a C if the symbol data contains colour or M if it is monochrome. **Type:** R **CE/SIZE:** A/1 **Value Range:** C, M **Subfield:** SRES2 **Name:** Reserved for future use. **Type:** R **CE/SIZE:** A/12 **Value Range:** 000000000000	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
SXSHDL	R/PVU	N/5	As specified in Table 6 for all profiles.	Calculated.	_X_ Same as Model Profile
SXSOFL	C	N/3	As specified in Table 6 for all profiles.	Calculated.	_X_ Same as Model Profile
SXSHD	C/PVU		As specified in Table 6 for all profiles.	Any tagged record extension specific to the symbol.	__ Same as Model Profile Select all that apply: ___ TREs are prohibited ___ Any Public TRE ___ Any Private TRE ___ Allowed Public TREs List: ___ Allowed Private TREs List: Specify constraints:

C.2.2.1.6 BIIF text subheader

Table C.6 is the proforma for profile specification of BIIF text subheader fields.

Table C.6 -- Text subheader

FIELD	TYPE	CE/ SIZE	PROFILE OPTIONS & RULES	MODEL PROFILE	NEW PROFILE SPECIFICATION Profile Name:____________
TE	R	A/2	Always TE for all profiles.	TE	_X Same as Model Profile
TEXTID	R/PVU	A/10	Any BCS-A string is allowed. Profile may specify further constraints.	Any BCS-A string.	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
TXTDT	R	N/14	As specified in Table 7.	CCYYMMDDhhmmss	_X Same as Model Profile
TXTITL	R/PVU	U8/80	Any UTF-8 string is allowed. Profile may specify further constraints.	Any BCS-A string.	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
TSSEC	R/PVU	U8/167	As specified in Table C.2.	As specified in Table C.2.	_X Same as Model Profile
ENCRYP	R/PVU	A/1	Encryption Encryption codes and meaning can be defined by profile, however, 0 = not encrypted.	CLEVEL 01: 0 CLEVEL 02: 0 1	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
TXTFMT	R	A/3	Any BCS-A code is allowed. Profile may specify further constraints. Representative values are: STA - Standard BCS Any BCS characters are allowed in the text data field. UC2 - Standard UCS-2 UC4 - Standard UCS-4 UT1 - Standard UTF-1 UT8 - Standard UTF-8 For ISO 10646, specify the adopted form, implementation level, and subset. For other text formats, provide full specification of use.	CLEVEL 01: STA CLEVEL 02: STA UC4 with implementation level 3	Select ONE of the following: ___ Same as Model Profile ___ The following are profile specific constraints:
TXSHDL	R/PVV	N/5	As specified in Table 7.	Calculated.	_X_ Same as Model Profile
TXSOFL	C	N/3	As specified in Table 7.	Calculated.	_X_ Same as Model Profile
TXSHD	C/PVU		As specified in Table 7.	Any tagged record extension specific to the text segment.	__ Same as Model Profile Select all that apply: ___ TREs are prohibited ___ Any Public TRE ___ Any Private TRE ___ Allowed Public TREs List: ___ Allowed Private TREs List: Specify constraints:

C.2.2.1.7 BIIF tagged record extensions (TREs)

The use and registration of BIIF tagged record extensions is not tied to any specific BIIF profile registration action. A BIIF profile can designate if none, any, or specific tagged record extensions are allowed within BIIF files conforming to the specific profile. No specific tagged record extensions are specified for use within the model profile. The model profile allows for the inclusion of any tagged record extension.

Submission of tagged record extensions for registration shall as a minimum use the proforma of Table C.7. The submission shall include additional documentation as needed to fully describe the proposed extension.

Table C.7 -- Tagged record extensions

FIELD	TYPE	CE/ SIZE	OPTIONS & RULES	NAME/DESCRIPTION	VALUE RANGE SPECIFICATION
TRETAG	R	A/6	Unique name not already registered as specified in Table 8.	Unique extension type identifier.	
TREL	R	N/5	As specified in Table 8.	Length of TREDATA field.	
TREDATA	R	As specified by TREL	Extend this table to fully define the contents of the data field. Provide additional narrative if needed to provide a comprehensive description of the extension data.	Extension data.	

C.2.2.1.8 BIIF data extension segments

The use and registration of BIIF data extension segments is not tied to any specific BIIF profile registration action. A BIIF profile can designate whether none, any, or specific data extension segments are allowed within BIIF files conforming to the specific profile. Two specific data extension segments are specified for use within the model profile. They are: TRE_OVERFLOW and TRANSPORTABLE_FILE_STRUCT.

Submission of data extension segments for registration shall as a minimum use the proforma of Table C.8a. The submission shall include additional documentation as needed to fully describe the proposed data extension segment.

Table C.8a -- Data extension segment proforma

FIELD	TYPE	CE/ SIZE	OPTIONS & RULES	NAME/DESCRIPTION	VALUE RANGE SPECIFICATION
DE	R	A/2	Always DE.	File part type.	DE
DESID	R	A/25	Unique BCS-A string not already registered.	Unique DES type identifier.	
DESVER	R	N/2	Version identifier.	Version of the DESTAG.	
DESSEC	R	U8/167	Not defined when registering the DES. The structure and parameters shall be those specified in Table C.2 for the BIIF profile of the file in which the DES appears.	Security specific parameters.	Not defined when registering the DES. The structure and parameters shall be those specified in Table C.2 for the BIIF profile of the file in which the DES appears.
DESOFLW	C/PVV	A/6	As specified in Table 10.	Overflowed header type.	When field is present: UDHD UDID XHD IXSHD SXSHD TXSHD
DESITEM	C	N/3	As specified in Table 10.	Data item reference.	000 - 999
DESSHL	R	N/4	As specified in Table 10.	Length of profile defined subheader fields.	0000 - 9999
DESSHF	C	Specified in DESSHL	Extend this table to fully define each subfield applicable to the DES.	Profile defined subheader fields.	
DESDATA	R/PVU	Per value from file header	Extend this table to fully define the contents of the data field. Provide additional narrative if needed to provide a comprehensive description of the DES.	Profile defined data field.	

C.2.2.1.9 BIIF reserved extension segments

The use and registration of BIIF reserved extension segments is not tied to any specific BIIF profile registration action. A BIIF profile can designate whether none, any, or specific reserved extension segments are allowed within BIIF files conforming to the specific profile. No specific reserved extension segments are specified for use within the model profile.

Submission of reserved extension segments for registration shall as a minimum use the proforma of Table C.8b. The submission shall include additional documentation as needed to fully describe the proposed reserved extension segment.

Table C.8b -- Reserved extension segment proforma

FIELD	TYPE	CE/ SIZE	OPTIONS & RULES	NAME/DESCRIPTION	VALUE RANGE SPECIFICATION
RE	R	A/2	Always RE.	File part type.	RE
RESID	R	A/25	Unique BCS-A string not already registered.	Unique RES type identifier.	
RESVER	R	N/2	Version identifier.	Version of the RESTAG.	
RESSEC	R	U8/167	Not defined when registering the RES. The structure and parameters shall be those specified in Table C.2 for the BIIF profile of the file in which the RES appears.	Security specific parameters.	Not defined when registering the RES. The structure and parameters shall be those specified in Table C.2 for the BIIF profile of the file in which the RES appears.
RESSHL	R	N/4	As specified for DES in Table 12.	Length of profile defined subheader fields.	0000 - 9999
RESSHF	C/PVU	Speci-fied in RESSHL	Extend this table to fully define each subfield applicable to the RES.	Profile defined subheader fields.	
RESDATA	R/PVU	Per value from file header	Extend this table to fully define the contents of the data field. Provide additional narrative if needed to provide a comprehensive description of the RES.	Profile defined data field.	

C.2.2.1.10 TFS Profile Proforma

The rules for defining valid profiles for TFS have the following objectives:

1. Provide interoperability between applications by specifically defining constraints on the TFS.
2. Provide a framework for developing TFS profiles.
3. Provide a BIIF TFS model profile for use with a BIIF Model Profile.
4. Set uniform rules for the development of conformance tests.

The BIIF TFS model profile is a usable instance of the TFS. The BIIF TFS model profile is a starting point from which an application-specific TFS profile should be defined. When creating a TFS profile, consideration should be given for each of the BIIF Model Profile specifications and either accept the specification or modify them only when they are not adequate.

The following table defines the BIIF TFS Model Profile and provides a mechanism for an application to define a unique TFS profile for conformance. The first column indicates the entry number for compliance, the second column is the requirement or TFS command stated for compliance. The third column is the BIIF TFS Model Profile. Each condition is checked as Required, Optional, Conditional, or Prohibited. The fourth column is provided for the registration of new TFS profiles. The boxes must be checked appropriately and all restrictions for the given command must be explained.

Table C.9 -- TFS profile proforma table

Table Entry Number	TFS Command or Specification	BIIF TFS Model Profile	New Profile Specification
1	Profile name	BIIF-TFS	Must be unique name not already registered.
2	BEGIN TFS	Required: X Optional: _ Conditional: _ Prohibited: _ Restrictions: Must be only one instance in metafile and be the first metafile command. The BEGIN TFS parameter shall contain the following data, delimited by the "/" character: Profile=profile_name BIIF-MODEL-TFS for model profile, Date=CCYYMMDDhhmmss, and Name="name of TFS". (case insensitive)	Same as model profile: X Profile must include: profile=profile_name, date=CCYYMMDDhhmmss, and name="name of TFS" delimited by the / character (case insensitive):
3	TFS VERSION	Required: X Optional: _ Conditional: _ Prohibited: _ Restrictions: Must be only one instance per metafile and be the second metafile command.	Same as model profile: X
4	TFS SECURITY	Required: X Optional: _ Conditional: _ Prohibited: _ Restrictions: Must be only one instance per metafile and be the third metafile command. The format must match the security used in the BIIF Model Profile.	Same as model profile: _ Required: X Optional: _ Conditional _ Prohibited: _ Restrictions: The format used for security must match the format used for the BIIF profile.

Table C.9 -- TFS profile proforma table *(continued)*

Table Entry Number	TFS Command or Specification	BIIF TFS Model Profile	New Profile Specification
5	TFS METADATA	Required: X Optional: _ Conditional: _ Prohibited: _ Restrictions: Must describe clearly the contents or purpose of the TFS and its transports contents.	Same as model profile: _ Required: _ Optional: _ Conditional: _ Prohibited: _ Restrictions:
6	TFS INDEX	Required: _ Optional: X Conditional: _ Prohibited: _ Restrictions: When used, it contains each transport name with the byte offset from the beginning of the file to each BEGIN TRANSPORT.	Same as model profile: _ Required: _ Optional: _ Conditional: _ Prohibited: _ Restrictions:
7	TFS SUBSCRIPTION	Required: _ Optional: X Conditional: _ Prohibited: _ Restrictions: Used for requesting TFS to be delivered to user determined by the TFS SUBSCRIPTION parameters. Each object delivery can be defined. Example: XXX=1 defines that object XXX be delivered in the TFS file. There are three subscription options: 1=pull requested data, 2=subscribe to requested data, and 3=cancel subscription.	Same as model profile: _ Required: _ Optional: _ Conditional: _ Prohibited: _ Restrictions:
8	TFS CONFIGURA-TION	Required: _ Optional: X Conditional: X Prohibited: _ Restrictions: Used for providing TFS information or requesting TFS information. When requesting TFS information the TFS CONFIG DATA command must be present.	Same as model profile: _ Required: _ Optional: _ Conditional: _ Prohibited: _ Restrictions:

Table C.9 -- TFS profile proforma table *(continued)*

Table Entry Number	TFS Command or Specification	BIIF TFS Model Profile	New Profile Specification
9	TFS CONFIG DATA	Required: _ Optional: X Conditional: X Prohibited: _ Restrictions: This command is conditionally required when the TFS CONFIGURATION contains 0 to indicate that configuration data is included. This command is optional otherwise. The TFS CONFIGURATION consists of OBJECT_NAME-SERVICE. For example, YYY=HTML. This indicates that an YYY object is of type HTML. It is up to the application to provide the Web Browser to read YYY.	Same as model profile: _ Required: _ Optional: _ Conditional: _ Prohibited: _ Restrictions:
10	TFS Descriptor Commands	Restrictions: All TFS Descriptor Commands shall start after the BEGIN TFS command and end before the BEGIN TRANSPORT command.	Same as model profile: X
11	BEGIN TRANSPORT	Required: _ Optional: X Conditional: X Prohibited: _ Restrictions: Optionally required when one or more profiles are to be transported. The BEGIN TRANSPORT parameter should include the name of the recipient.	Same as model profile: X
12	TRANSPORT SECURITY	Required: _ Optional: _ Conditional: X Prohibited: _ Restrictions: Conditionally required when BEGIN TRANSPORT command is present. Must be only one instance per transport and be the first command after the BEGIN TRANSPORT command. The format must match the security used in the BIIF Model Profile.	Same as model profile: _ Required: _ Optional: _ Conditional: X Prohibited: _ Restrictions: Conditionally required when BEGIN TRANSPORT command is present and be the first metafile command after BEGIN TRANSPORT command. The format used for security must match the format used for the BIIF profile.

Table C.9 -- TFS profile proforma table *(continued)*

Table Entry Number	TFS Command or Specification	BIIF TFS Model Profile	New Profile Specification
13	TRANSPORT METADATA	Required: X Optional: _ Conditional: _ Prohibited: _ Restrictions: Must describe clearly the contents or purpose of the transport and its contents.	Same as model profile: _ Required: _ Optional: _ Conditional: _ Prohibited: _ Restrictions:
14	TRANSPORT INDEX	Required: _ Optional: X Conditional: _ Prohibited: _ Restrictions: When used, it contains each profile name with the byte offset from the beginning of the BEGIN TRANSPORT command to each high level BEGIN PROFILE.	Same as model profile: _ Required: _ Optional: _ Conditional: _ Prohibited: _ Restrictions:
15	Transport Descriptor Commands	Restrictions: All Transport Descriptor Commands for the given transport shall start after the BEGIN TRANSPORT command and end before the BEGIN TRANSPORT BODY command.	Same as model profile: X
16	BEGIN TRANSPORT BODY	Required: _ Optional: _ Conditional: X Prohibited: _ Restrictions: Conditionally required for each BEGIN TRANSPORT command. It must be the first command after the Transport Descriptor Commands.	Same as model profile: X
17	BEGIN PROFILE	Required: _ Optional: X Conditional: X Prohibited: _ Restrictions: Conditionally required when PROFILE ACTION or PROFILE OBJECT commands are present within the given profile state.	Same as model profile: X

Table C.9 -- TFS profile proforma table *(continued)*

Table Entry Number	TFS Command or Specification	BIIF TFS Model Profile	New Profile Specification
18	PROFILE SECURITY	Required: X Optional: _ Conditional: X Prohibited: _ Restrictions: Conditionally required when BEGIN PROFILE command is present. Must be only one instance per profile and be the first command after the BEGIN PROFILE command. The format must match the security used in the BIIF Model Profile.	Same as model profile: _ Required: X Optional: _ Conditional: _ Prohibited: _ Restrictions: The format used for security must match the format used for the BIIF profile.
19	PROFILE METADATA	Required: X Optional: _ Conditional: _ Prohibited: _ Restrictions: Must describe clearly the contents or purpose of the profile and its contents.	Same as model profile: _ Required: _ Optional: _ Conditional: _ Prohibited: _ Restrictions:
20	PROFILE INDEX	Required: _ Optional: X Conditional: _ Prohibited: _ Restrictions: When used, it contains each PROFILE ACTION, PROFILE OBJECT, or an embedded BEGIN PROFILE name with the byte offset from the beginning of the BEGIN PROFILE command to each next level command.	Same as model profile: _ Required: _ Optional: _ Conditional: _ Prohibited: _ Restrictions:
21	BEGIN PROFILE BODY	Required: _ Optional: _ Conditional: X Prohibited: _ Conditionally required for each BEGIN PROFILE command. It must be the first command after the Profile Descriptor Commands.	Same as model profile: X
22	Profile Descriptor Commands	Restrictions: All Profile Descriptor Commands for the given profile shall start after the BEGIN PROFILE command and end before the BEGIN PROFILE BODY command.	Same as model profile: X

Table C.9 -- TFS profile proforma table *(continued)*

Table Entry Number	TFS Command or Specification	BIIF TFS Model Profile	New Profile Specification
23	PROFILE ACTION	Required: _ Optional: X Conditional: _ Prohibited: _ Restrictions: When used parameter must be in the form "OBJECT_NAME= action". This command must be ignored and documented if not known to the application.	Same as model profile: _ Required: _ Optional: _ Conditional: _ Prohibited: _ Restrictions:
24	PROFILE OBJECT	Required: _ Optional: X Conditional: _ Prohibited: _ Restrictions: Contains OBJECT_TYPE 1or 2 OBJECT_NAME, OBJECT_STORAGE 0 or 1 or 2, and OBJECT_DATA if OBJECT_STORAGE is not 0. Profile objects include all PIKS1 (PIKS Foundation) objects excluding the PIKS_IMAGE object. A PIKS image object at Foundation Level does not require the TFS mechanism.	Same as model profile: _ Required: _ Optional: _ Conditional: _ Prohibited: _ Restrictions:
25	END PROFILE	Required: _ Optional: _ Conditional: X Prohibited: _ Restrictions: Must be only one instance per profile state and be the last command in the profile state.	Same as model profile: X
26	END TRANSPORT	Required: _ Optional: _ Conditional: X Prohibited: _ Restrictions: Must be only one instance per transport state and be the last command in the transport state.	Same as model profile: X
27	END TFS	Required: X Optional: _ Conditional: _ Prohibited: _ Restrictions: Must be only one instance per metafile and be the last metafile command.	Same as model profile: X
28	TFS ESCAPE	Required: _ Optional: _ Conditional: _ Prohibited: X Restrictions: All TFS ESCAPE commands are ignored.	Same as model profile: _ Required: _ Optional: _ Conditional: _ Prohibited: _ Restrictions:

Table C.9 -- TFS profile proforma table *(concluded)*

Table Entry Number	TFS Command or Specification	BIIF TFS Model Profile	New Profile Specification
29	Nested Profiles	Restrictions: Nested profiles or profiles within profiles are allowed. Nested levels allowed: 10	Same as model profile: _ Restrictions:
30	Number of Objects	Restrictions: There is no restriction to the number of objects as long as the file sizes of the TFS and BIIF comply within the certification conformance level.	Same as model profile: _ Restrictions:
31	Number of Profiles	Restrictions: There is no restriction to the number of profiles as long as the file sizes of the TFS and BIIF comply within the certification conformance level.	Same as model profile: _ Restrictions:
32	Number of Transports	Restrictions: There is no restriction to the number of transports as long as the file sizes of the TFS and BIIF comply within the certification conformance level.	Same as model profile: _ Restrictions:

C.2.2.1.11 Implementation Support Requirements

Table C.10 is the proforma for specification of implementation support requirements as called for in Clause 5.4. Note that this proforma does not specify content specifics of BIIF field values as done in other proforma tables. It identifies implementation specific requirements beyond format content.

Table C.10 – Implementation Support Requirements

Profile Options and Rules	Model Profile	New Profile Specification Profile Name: _________
As specified in 5.4, requirements shall be expressed in a manner that is testable.	1. The Model Profile imposes no further constraints for: - Image productive fidelity - Compression/quality goals - Content transforms Special handling or processing rates or application semantics. 2. An implementation capable of producing BIIF Model Profile compliant files shall ensure that all such producer files are fully within the constraints of the applicable complexity level of the model profile. A producer need not support the full extent of allowable options within the profile. 3. An implementation capable of interpreting the Model Profile may implement complexity level 01 only or both complexity levels 01 and 02. The implementation shall be able to interpret and use any model profile compliant file for the complexity level implemented.	Select one or more of the following: - Same as Model Profile - The following are profile specific constraints:

Annex D
(informative)

Implementation Considerations and Product Configurations

D.1 Implementation considerations and product configurations

The Basic Imagery Interchange Format (BIIF) has been developed to provide image exchange capabilities and interoperability among computer systems of various designs and capabilities. For the purposes of BIIF, interoperability means the ability to exchange BIIF formatted imagery products among BIIF capable systems in a manner that is meaningful and useful to the end users. This annex will discuss general considerations pertinent to successful implementation of BIIF. Guidelines will be presented, and potential problems will be highlighted. BIIF preprocessor and postprocessor software, the software potentially necessary to write and read a BIIF file based on host native mode files containing the data items to be included, are to be written by the user. The combination of the preprocessor and postprocessor hereafter will be referred to as the "BIIF implementation." Preprocessing is sometimes called "packing," and postprocessing is called "unpacking."

Subclause D.4 discusses several generalized product configurations that can be used as the basis for defining specific imagery products. These product configurations are typical of those successfully used within the imagery and mapping community to date.

D.2 TRE_OVERFLOW example

The following is a description of a sample instance that uses a TRE_OVERFLOW Data Extension Segment (DES). For this example, the BIIF file consists of a file header, a single image segment and a single data extension segment. Additionally, there are four TREs which relate to the image segment. The four TREs should be placed in the image subheader extension fields (UDID and IXSHD). In this example, the four TREs have the following total lengths:

TRE1 - 91,000 bytes
TRE2 - 32,000 bytes
TRE3 - 27,000 bytes
TRE4 - 42,000 bytes

The maximum capacity of either UDID or IXSHD is 99,996 bytes. Hence the placement of TREs for this example is as follows:

TRE1is placed in the UDID field. (Uses 91,000 of available 99,996 bytes. No other TRE is small enough to be added to the UDID field in its entirety. The first 3 bytes of the UDID field must contain "000" to indicate no overflow to a DES.) TRE2 and TRE3 are placed in the IXSHD field. (Uses 59,000 of the available 99,999 bytes.) TRE4 is placed in the TRE_OVERFLOW DES since there is not enough space remaining in IXSHD to contain the entire TRE.) The first three bytes of the IXSHD field must have the value 001 to indicate that overflow is going into the first DES in the file. The first DES is named "TRE_OVERFLOW". The DESOFLW field contains IXSHD to show the field being overflowed. The DESITEM field contains 001 to show it is the first image segment from which the IXSHD field is overflowed.

D.3 Scope of implementation

BIIF describes the format of images, graphics, text, and associated metadata within the BIIF file only. It does not define the image, graphics, or text requirements of the host system. The host system is responsible for the handling of unpacked image and text, as well as image, graphics, and text display capabilities.

D.3.1 Creating headers and subheaders

This standard and associated profiles specify legal values for the header and subheader fields. The BIIF implementation for any particular host system will be responsible for enforcing the field values when creating files as stated in this standard.

D.3.2 Character counts

BIIF uses explicit byte counts to delimit fields. No end-of-field characters are used. These byte counts are critical for the proper interpretation of a BIIF file. The BIIF implementation should compute these byte counts based on the file contents to insure accuracy. All fields in BIIF header and subheaders must be present exactly as specified in BIIF header and subheader descriptions, and no additional fields may be inserted. BIIF uses various conditional fields whose presence, length, and structure is determined by previous fields and counts. If an expected conditional field is missing, the remainder of the file may be misinterpreted. A similar result will occur if a conditional field is inserted when it is not required. For these reasons, the item count fields are critical, and every effort must be made to ensure their accuracy.

D.3.3 Data entry

To reduce any operator workload imposed by the implementation, each implementation should provide where plausible for the automatic entry of data. Global default values for the particular BIIF version should be inserted automatically in the file. System default values, such as the standard size parameters for a base image, also should be entered automatically by the implementation. Values that are known to the system, such as the time or the computed size of an overlay, also should be entered automatically. Where operator value selection is needed, the use of pre-defined selection lists of valid values is encouraged. Range or value checking logic for operator input field entries will help avoid inadvertent operator entry errors.

D.3.4 Tagged Record Extensions

Users may need to add additional data to a BIIF file header or image subheader. To accommodate this requirement, user-defined data and extension fields are provided in the file header and segment subheader. One potential use for the user-defined data and extension subheader data is to provide space for directly associating acquisition parameters with the image. Use of these fields requires insertion of tagged records that implement the extension as described in this standard. Public tags shall be registered with the BIIF registration authority according to procedures available from the authority. This procedure ensures that different users will not use the same tag to flag different extended data. It also provides for configuration management of Tagged Record Extension formats where the extended data are expected to be used by a wide audience of users.

D.3.5 Out-of-bounds field values

The file creator is responsible for ensuring that all BIIF field values are within the bounds specified by BIIF document. An out-of-bounds value in a BIIF field indicates that either an error occurred or that the sending station was not in full compliance with BIIF.

D.3.6 Use of images

BIIF specifies a format for images contained within a BIIF file only. A BIIF implementation must be capable of translating this format to and from the host system's local format. Some host systems have multiple formats for binary data. In these cases, the BIIF implementation must use the appropriate host format to provide the necessary data exchange services with other system packages. When imagery data of less than M bits-per-pixel is displayed on an M-bit (2^M gray shades) display device, it must be transformed into the dynamic range of the device. One way to do this is to modify the LUTs of the display device. However, if M-bit and less than M-bit imagery is displayed simultaneously, the M-bit image will appear distorted. The recommended method is to convert the less than M-bit imagery into M-bit imagery, then use the standard LUTs. The following equation will transform a less than M-bit pixel into and M-bit pixel:

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

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

D.3.7 Use of text files

The text format field is provided to help the reader of the file determine how to interpret the text data received. The file reader is responsible for interpreting the various text formats. Format designations explicitly supported by BIIF are as follows:

D.3.7.1 BCS (TXTFMT=STA)

BIIF BCS is a specified format to provide a common format for all BIIF implementations. The BCS code shall be represented as depicted in Tables D-1. This is the BCS code (Basic Latin Set) represented in ISO 646. The BCS codes shall be eight bits, a_1 through a_8. The eighth bit, a_8, shall be set to 0. A_8 shall be the Most Significant Bit (MSB), and a_1 shall be the Least Significant Bit (LSB). It is intended to provide for simple communications among BIIF stations. BIIF BCS-A format is comprised of the following BCS characters (all numbers are decimal): Line Feed (10), Form Feed (12), Carriage Return (13), and space (32) through tilde (126). This set includes all the alphanumeric characters as well as all commonly used punctuation characters. All lines within a BIIF STA file will be separated by carriage return/line feed pairs. It is the responsibility of the local system to translate these pairs into the local format. BIIF BCS has no standard line length. The host system must be capable of processing lines that are longer than the local standard.

D.3.7.2 Additional TXTFMT Codes

BIIF allows multi-octet codes to be contained in the text data field. Different systems interpret these codes for various purposes. The code "STA" in the TXTFMT field shall mean that only BCS-A characters appear in the text data field. A

TXTFMT value of UC2 indicates UCS-2, two octet UCS , UC4 is UCS-4, four octet UCS, UT1 is UTF-1 (Basic Latin and Latin-1 Supplement, Tables D1-D4), UCS Transformation Format 1, and UT8 is UTF-8, UCS Transformation Format 8 (Amendment 2 of ISO 10646).

Table D.1 -- Basic Latin character set

	000	001	002	003	004	005	006	007
0	000	016	SP 032	0 048	@ 064	P 080	` 096	p 112
1	001	017	! 033	1 049	A 065	Q 081	a 097	q 113
2	002	018	" 034	2 050	B 066	R 082	b 098	r 114
3	003	019	# 035	3 051	C 067	S 083	c 099	s 115
4	004	020	$ 036	4 052	D 068	T 084	d 100	t 116
5	005	021	% 037	5 053	E 069	U 085	e 101	u 117
6	006	022	& 038	6 054	F 070	V 086	f 102	v 118
7	007	023	' 039	7 055	G 071	W 087	g 103	w 119
8	008	024	(040	8 056	H 072	X 088	h 104	x 120
9	009	025	) 041	9 057	I 073	Y 089	i 105	y 121
A	010	026	* 042	: 058	J 074	Z 090	j 106	z 122
B	011	027	+ 043	; 059	K 075	[091	k 107	{ 123
C	012	028	, 044	< 060	L 076	\ 092	l 108	| 124
D	013	029	- 045	= 061	M 077	] 093	m 109	} 125
E	014	030	. 046	> 062	N 078	^ 094	n 110	~ 126
F	015	031	/ 047	? 063	O 079	_ 095	o 111	128

Table D.2 -- Basic Latin character set explanation

Decimal	Hex	Name
032	20	SPACE
033	21	EXCLAMATION MARK
034	22	QUOTATION MARK
035	23	NUMBER SIGN
036	24	DOLLAR SIGN
037	25	PERCENT SIGN
038	26	AMPERSAND
039	27	APOSTROPHE
040	28	LEFT PARENTHESIS
041	29	RIGHT PARENTHESIS
042	2A	ASTERISK
043	2B	PLUS SIGN
044	2C	COMMA
045	2D	HYPHEN-MINUSb
046	2E	FULL STOP
047	2F	SOLIQUS
048	30	DIGIT ZERO
049	31	DIGIT ONE
050	32	DIGIT TWO
051	33	DIGIT THREE
052	34	DIGIT FOUR
053	35	DIGIT FIVE
054	36	DIGIT SIX
055	37	DIGIT SEVEN
056	38	DIGIT EIGHT
057	39	DIGIT NINE
058	3A	COLON
059	3B	SEMICOLON
060	3C	LESS-THAN SIGN
061	3D	EQUALS SIGN
062	3E	GREATER-THAN SIGN
063	3F	QUESTION MARK
064	40	COMMERCIAL AT
065	41	LATIN CAPITAL LETTER A
066	42	LATIN CAPITAL B
067	43	LATIN CAPITAL C
068	44	LATIN CAPITAL D
069	45	LATIN CAPITAL E
070	46	LATIN CAPITAL F
071	47	LATIN CAPITAL G
072	48	LATIN CAPITAL H
073	49	LATIN CAPITAL I
074	4A	LATIN CAPITAL J
075	4B	LATIN CAPITAL K
076	4C	LATIN CAPITAL L
077	4D	LATIN CAPITAL M
078	4E	LATIN CAPITAL N
079	4F	LATIN CAPITAL O
080	50	LATIN CAPITAL P
081	51	LATIN CAPITAL Q

Table D.2 -- Basic Latin character set explanation *(concluded)*

Decimal	Hex	Name
082	52	LATIN CAPITAL R
083	53	LATIN CAPITAL S
084	54	LATIN CAPITAL T
085	55	LATIN CAPITAL U
086	56	LATIN CAPITAL V
087	57	LATIN CAPITAL W
088	58	LATIN CAPITAL X
089	59	LATIN CAPITAL Y
090	5A	LATIN CAPITAL Z
091	5B	LEFT SQUARE BRACKET
092	5C	REVERSE SOLIDUS
093	5D	RIGHT SQUARE BRACKET
094	5E	CIRCUMFLEX ACCENT
095	5F	LOW LINE
096	60	GRAVE ACCENT
097	61	LATIN SMALL LETTER A
098	62	LATIN SMALL LETTER B
099	63	LATIN SMALL LETTER C
100	64	LATIN SMALL LETTER D
101	65	LATIN SMALL LETTER E
102	66	LATIN SMALL LETTER F
103	67	LATIN SMALL LETTER G
104	68	LATIN SMALL LETTER H
105	69	LATIN SMALL LETTER I
106	6A	LATIN SMALL LETTER J
107	6B	LATIN SMALL LETTER K
108	6C	LATIN SMALL LETTER L
109	6D	LATIN SMALL LETTER M
110	6E	LATIN SMALL LETTER N
111	6F	LATIN SMALL LETTER O
112	70	LATIN SMALL LETTER P
113	71	LATIN SMALL LETTER Q
114	72	LATIN SMALL LETTER R
115	73	LATIN SMALL LETTER S
116	74	LATIN SMALL LETTER T
117	75	LATIN SMALL LETTER U
118	76	LATIN SMALL LETTER V
119	77	LATIN SMALL LETTER W
120	78	LATIN SMALL LETTER X
121	79	LATIN SMALL LETTER Y
122	7A	LATIN SMALL LETTER Z
123	7B	LEFT CURLY BRACKET
124	7C	VERTICAL LINE
125	7D	RIGHT CURLY BRACKET
126	7E	TILDE

Table D.3 -- Latin-1 supplement character set

	008	009	00A	00B	00C	00D	00E	00F
0	128	144	NB SP 160	° 176	À 192	Ð 208	à 224	ð 240
1	129	145	¡ 161	± 177	Á 193	Ñ 209	á 225	ñ 241
2	130	146	¢ 162	² 178	Â 194	Ò 210	â 226	ò 242
3	131	147	£ 163	³ 179	Ã 195	Ó 211	ã 227	ó 243
4	132	148	¤ 164	´ 180	Ä 196	Ô 212	ä 228	ô 244
5	133	149	¥ 165	µ 181	Å 197	Õ 213	å 229	õ 245
6	134	150	¦ 166	¶ 182	Æ 198	Ö 214	æ 230	ö 246
7	135	151	§ 167	· 183	Ç 199	× 215	ç 231	÷ 247
8	136	152	¨ 168	¸ 184	È 200	Ø 216	è 232	ø 248
9	137	153	© 169	¹ 185	É 201	Ù 217	é 233	ù 249
A	138	154	ª 170	º 186	Ê 202	Ú 218	ê 234	ú 250
B	139	155	« 171	» 187	Ë 203	Û 219	ë 235	û 251
C	140	156	¬ 172	¼ 188	Ì 204	Ü 220	ì 236	ü 252
D	141	157	- 173	½ 189	Í 205	Ý 221	í 237	ý 253
E	142	158	® 174	¾ 190	Î 206	Þ 222	î 238	þ 254
F	143	159	¯ 175	¿ 191	Ï 207	ß_ 223	ï 239	ÿ 255

Table D.4 -- Latin-1 supplement character set explanation

Decimal	Hex	Name
160	A0	NO BREAK SPACE
161	A1	INVERTED EXCLAMATION MARK
162	A2	CENT SIGN
163	A3	POUND SIGN
164	A4	CURRENCY SIGN
165	A5	YEN SIGN
166	A6	BROKEN BAR
167	A7	SECTION SIGN
168	A8	DIAERESIS
169	A9	COPYRIGHT
170	AA	FEMININE ORDINAL INDICATOR
171	AB	LEFT-POINTING DOUBLE ANGLE QUOTATION MARK
172	AC	NOT SIGN
173	AD	SOFT HYPHEN
174	AE	REGISTERED SIGN
175	AF	MACRON
176	B0	DEGREE SIGN
177	B1	PLUS-MINUS SIGN
178	B2	SUPERSCRIPT TWO
179	B3	SUPERSCRIPT THREE
180	B4	ACUTE ACCENT
181	B5	MICRO SIGN
182	B6	PILCROW SIGN
183	B7	MIDDLE DOT
184	B8	CEDILLA
185	B9	SUPERSCRIPT ONE
186	BA	MASCULINE ORDINAL INDICATOR
187	BB	RIGHT POINTING DOUBLE ANGLE QUOTATION MARK
188	BC	VULGAR FRACTION ONE QUARTER
189	BD	VULGAR FRACTION ONE HALF
190	BE	VULGAR FRACTION THREE QUARTERS
191	BF	INVERTED QUESTION MARK
192	C0	LATIN CAPITAL LETTER A WITH GRAVE
193	C1	LATIN CAPITAL LETTER A WITH ACUTE
194	C2	LATIN CAPITAL LETTER A WITH CIRCUMFLEX
195	C3	LATIN CAPITAL LETTER A WITH TILDE
196	C4	LATIN CAPITAL LETTER A WITH DIAERESIS
197	C5	LATIN CAPITAL LETTER A WITH RING ABOVE
198	C6	LATIN CAPITAL LIGATURE AE
199	C7	LATIN CAPITAL LETTER C WITH CEDILLA
200	C8	LATIN CAPITAL LETTER E WITH GRAVE
201	C9	LATIN CAPITAL LETTER E WITH ACUTE
202	CA	LATIN CAPITAL LETTER E WITH CIRCUMFLEX
203	CB	LATIN CAPITAL LETTER E WITH DIAERESIS
204	CC	LATIN CAPITAL LETTER I WITH GRAVE
205	CD	LATIN CAPITAL LETTER I WITH ACUTE
206	CE	LATIN CAPITAL LETTER I WITH CIRCUMFLEX
207	CF	LATIN CAPITAL LETTER I WITH DIAERESIS
208	D0	LATIN CAPITAL LETTER ETH (ICELANDIC)
209	D1	LATIN CAPITAL N WITH TILDE
210	D2	LATIN CAPITAL LETTER O WITH GRAVE
211	D3	LATIN CAPITAL LETTER O WITH ACUTE
212	D4	LATIN CAPITAL LETTER O WITH CIRCUMFLEX

Table D.4 -- Latin-1 supplement character set explanation *(continued)*

Decimal	Hex	Name
213	D5	LATIN CAPITAL LETTER O WITH TILDE
214	D6	LATIN CAPITAL LETTER O WITH DIAERESIS
215	D7	MULTIPLICATION SIGN
216	D8	LATIN CAPITAL LETTER WITH STROKE
217	D9	LATIN CAPITAL LETTER U WITH GRAVE
218	DA	LATIN CAPITAL LETTER U WITH ACUTE
219	DB	LATIN CAPITAL LETTER U WITH CIRCUMFLEX
220	DC	LATIN CAPITAL LETTER U WITH DIAERESIS
221	DD	LATIN CAPITAL LETTER Y WITH ACUTE
222	DE	LATIN CAPITAL LETTER THORN (ICELANDIC)
223	DF	LATIN SMALL LETTER SHARP S (GERMAN)
224	E0	LATIN SMALL A WITH GRAVE
225	E1	LATIN SMALL LETTER A WITH ACUTE
226	E2	LATIN SMALL LETTER A WITH CIRCUMFLEX
227	E3	LATIN SMALL LETTER A WITH TILDE
228	E4	LATIN SMALL LETTER A WITH DIAERESIS
229	E5	LATIN SMALL LETTER A WITH RING ABOVE
230	E6	LATIN SMALL LIGATURE AE
231	E7	LATIN SMALL LETTER C WITH CEDILLA
232	E8	LATIN SMALL LETTER E WITH GRAVE
233	E9	LATIN SMALL LETTER E WITH ACUTE
234	EA	LATIN SMALL LETTER E WITH CIRCUMFLEX
235	EB	LATIN SMALL LETTER E WITH DIAERESIS
236	EC	LATIN SMALL LETTER I WITH GRAVE
237	ED	LATIN SMALL LETTER I WITH ACUTE
238	EE	LATIN SMALL LETTER I WITH CIRCUMFLEX
239	EF	LATIN SMALL LETTER I WITH DIAERESIS
240	F0	LATIN SMALL LETTER ETH (ICELANDIC)
241	F1	LATIN SMALL LETTER N WITH TILDE
242	F2	LATIN SMALL LETTER O WITH GRAVE
243	F3	LATIN SMALL LETTER O WITH ACUTE
244	F4	LATIN SMALL LETTER O WITH CIRCUMFLEX
245	F5	LATIN SMALL LETTER O WITH TILDE
246	F6	LATIN SMALL LETTER O WITH DIAERESIS
247	F7	DIVISION SIGN
248	F8	LATIN SMALL LETTER O WITH STROKE
249	F9	LATIN SMALL LETTER U WITH GRAVE
250	FA	LATIN SMALL LETTER U WITH ACUTE
251	FB	LATIN SMALL LETTER U WITH CIRCUMFLEX
252	FC	LATIN SMALL LETTER U WITH DIAERESIS
253	FD	LATIN SMALL LETTER Y WITH ACUTE
254	FE	LATIN SMALL LETTER THORN (ICELANDIC)
255	FF	LATIN SMALL LETTER Y WITH DIAERESIS

D.3.8 File system constraints

A BIIF file is presented as a stream of contiguous octets. The file length value which appears in the file header represents an exact count of the meaningful octets contained within the BIIF file structure. This value may not always correspond with file size values reported by some computer file storage systems (e.g. those that store files on multi-byte block boundaries vice single byte boundaries). The requirement for storing and exchanging BIIF files between systems with differing file storage structures must be accommodated when implementing BIIF. When the file storage system reports a larger file size than the value contained in the file header, the file length value in the file is the basis for determining the last byte of meaningful data within the file. When a smaller file size is reported by the file system, it is an indication that the file has been inadvertently truncated during the file exchange process and is therefore incomplete or otherwise defective.

D.3.9 Security considerations

A BIIF file contains sufficient security information in the file header, image and graphic subheaders to allow implementors to meet virtually any security requirement for controlling the use and presentation of data. Exact security information handling requirements generally are specified by appropriate accreditation authorities or specific user requirements. It is suggested that implementors extract the appropriate security markings from one or more of the header/subheaders and ensure that the information is always displayed whenever the pertinent part of BIIF file is displayed.

D.4 Product configurations

BIIF provides a very flexible means to package imagery products. Interoperability places a significant burden on BIIF read capable implementations to be able to interpret and use potentially any combination and permutation of BIIF file format options that may be created by BIIF file producers. Consequently, significant care should be taken when defining product specifications for BIIF formatted imagery products.

The objective of the following discussion is to describe several generalized product configurations that can be used as the basis for defining specific imagery products. These product configurations are typical of those successfully used within the imagery and mapping community to date.

D.4.1 General

An imagery product may potentially be produced under one of the following concepts:

1. Single File, Single Base Image - This is the most common use of BIIF format. In this product concept, the BIIF file is produced with a focus on a single image, commonly called the ‘base image’. All other segments and extended data within the file are focused on amplifying the information portrayed in the base image.
2. Single File, Multiple Images - In this product concept, the BIIF file is produced containing multiple images, all of which have equal or similar significance to the value of the product. Other segments and extended data within the file are focused on amplifying the information portrayed in the image(s) to which they are associated.
3. Single File, No Image - This type of product may only have graphic segments, or only text segments, or only extension segments, or any combination of these segments. The significance of the data within the file may pertain only to that file, or it may pertain to one or more files with which it is associated.
4. Multiple Correlated Files - For this product concept, the product is comprised of multiple BIIF files that are interrelated as explicitly defined in the product specification.

D.4.1.1 Single file, single base image

For this type of product file, there is one image of central focus, the base image, placed on the Common Coordinate System (CCS) plane. Its first pixel may be located at the origin (0,0) of the CCS or off-set from the CCS origin according to the row/column coordinate values placed in the location (LOC) field of the image subheader. Figure D.1 provides a representative portrayal for the following discussion.

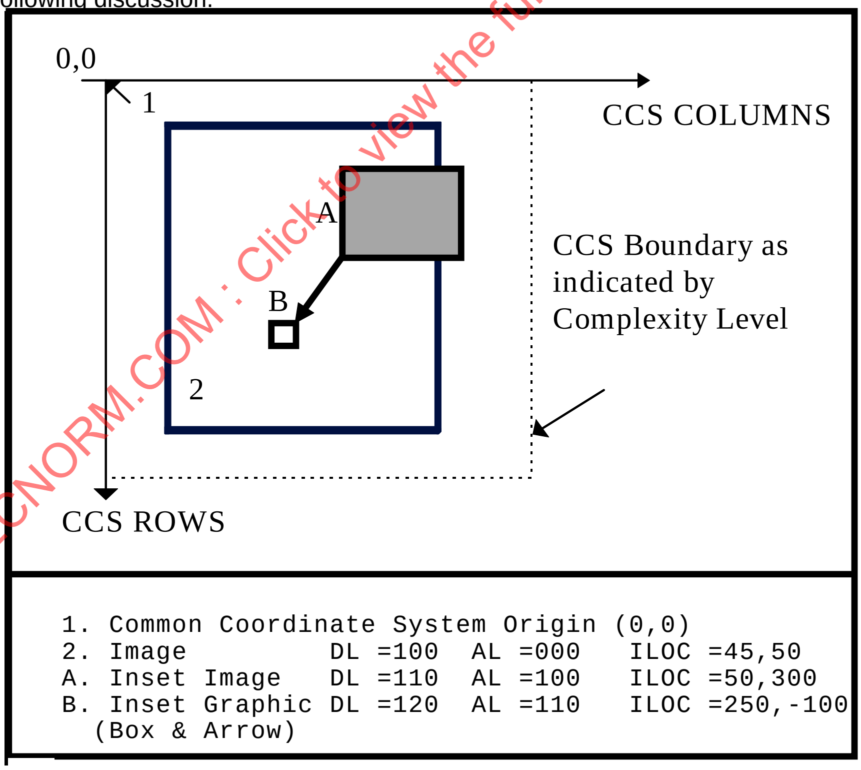


Figure D.1 -- Single file, single base image representation

D.4.1.1.1 Image segment overlays

Additional images, often called subimages or inset images, may be included as separate image segments in the file. The purpose of these images is to add information or clarity about the base image. Their placement in the CCS plane is controlled by the value of each segment’s Attachment Level (AL) and Location (LOC) row/column value. When overlay images are attached to the base image, the LOC value represents a row/column off-set in the CCS from the location specified by the base image row/column LOC value. If the overlay image is unattached to any other segment (AL=000), the overlay's LOC value is a row/column off-set from the CCS origin (0,0).

D.4.1.1.2 Graphic segment overlays

Graphic Segments are used to provide graphical (lines, polygons, ellipses, etc.) and textual annotation to the base image. The graphic representation is done using Computer Graphics Metafile (CGM). In a manner similar to image segment overlays, the placement of graphics in the CCS plane is controlled by the value of each segment's AL and LOC values. CGM has its own internal coordinate space called "Virtual Display Coordinates (VDC)" that has its own defined origin (0,0) point. The row/column value in the graphic segment LOC field identifies the placement of the graphic's VDC origin point relative to the CCS origin when AL=000 or relative to the segment LOC to which it is attached.

D.4.1.1.3 Non-destructive overlays

BIIF image and graphic segment overlays are handled in a non-destructive manner. The overlays may be placed anywhere within the bounds of the CCS (defined for a specific BIIF file by the Complexity level (Clevel)). They may be placed totally on the base image, partially on the base image, or entirely off of the base image. Any image or graphic segment can be placed on or under any other segment, fully or partially. The visibility of pixel values of overlapping segments is determined by the Display Level (DL) assigned to that segment. Each displayable segment (images and graphics) is assigned a DL (ranging from 001 - 999) that is unique within the file. At any CCS pixel location shared by more than one image or graphic, the visible pixel value is the one from the segment having the greatest DL value. If the user of a BIIF file opts to move an overlay or turn off the presentation of an overlay, the next greatest underlying pixel value(s) will then become visible. This approach allows for the non-destructible nature of BIIF overlays as opposed to the 'burned in' approach where overlay pixel values are used to replace pixels values of the underlying image.

D.4.1.1.4 Text Segments

Text segments allow inclusion in BIIF file of textual information related to the base image, perhaps a textual description of the activities portrayed in the image.

D.4.1.1.5 Extension data

BIIF file header and each standard data type sub-header have designated expandable fields to allow for the optional inclusion of extension data. The inclusion of extension data provides the ability to add data/information about the standard data type (metadata) that is not contained in the basic fields of the headers and subheaders. The additional data is contained within one or more BIIF tagged record extensions that are placed in the appropriate field (user defined data field or extended data field) of the standard data type subheader for which the metadata applies. When tagged record extensions have application across multiple data types in the file, or otherwise apply to the entire BIIF file in general, they are placed in the appropriate file header fields.

Whereas general purpose BIIF readers should always be able to portray image and graphic segments and act on standard header and subheader data, they may not always be able to act on product specific extension data. Upon receipt of a file that contains extension data, a BIIF compliant system should at least ignore the extensions and properly interpret the other legal components of the BIIF file. Exemplary use of tagged record extensions:

1. Data about people, buildings, places, landmarks, equipment or other objects that may appear in the image
2. Data to allow correlation of information among multiple images and annotations within a BIIF file
3. Data about the equipment settings used to obtain the digital image, xray, etc.
4. Data to allow geopositioning of items in the imagery or measurement of distances of items in the imagery.

D.4.1.2 Single file, multiple images

For this type of product file, multiple images of equal or similar focus (multiple 'base' images) are placed within the Common Coordinate System (CCS) plan. Each image is located at an off-set from the CCS origin such that there is no overlap among the images. The Complexity Level of the file must be chosen such that the bounds of the CCS for the file are sufficient to contain the extent of all segments within the file. Figure D.2 provides a representative portrayal for this product type.