Machine Independent Format

MIF Data Elements

ERDAS IMAGINE uses the Machine Independent Format (MIF) to store data in a fashion which can be read by a variety of machines. This format provides support for converting data between the IMAGINE standard data format and that of the specific host's architecture. Files created using this package on one machine will be readable from another machine with no explicit data translation.

Each MIF file is made up of one or more of the data elements explained below.

EMIF_T_U1 (Unsigned 1−bit Integer)

U1 is for unsigned 1−bit integers (0 − 1). This data type can be used for bitmap images with "yes/no" conditions. When the data are read from a MIF file, they are automatically expanded to give one value per byte in memory. When they are written to the file, they are automatically compressed to place eight values into one output byte.

7	6	5	4	3	2	1	0
U1_7	U1_6	U1_5	U1_4	U1_3	U1_2	U1_1	U1_0

byte 0

EMIF_T_U2 (Unsigned 2−bit Integer)

U2 is for unsigned 2−bit integers (0 − 3). This data type can be used for thematic data with 4 or fewer classes. When the data are read from a MIF file they are automatically expanded to give one value per byte in memory. When they are written to the file, they are automatically compressed to place four values into one output byte.

7	5	3	1
U2_3	U2_2	U2_1	U2_0

byte 0

EMIF_T_U4 (Unsigned 4−bit Integer)

U4 is for unsigned 4−bit integers (0 − 15). This data type can be used for thematic data with 16 or fewer classes. When these data are read from a MIF file, they are automatically expanded to give one value per byte. When they are written to the file they are automatically compressed to place two values into one output byte.

7	3
U4_1	U4_0
byte 0	

EMIF_T_UCHAR (8−bit Unsigned Integer)

This stores an 8−bit unsigned integer. It is most typically used to store characters and raster imagery

7
integer
byte 0

.

EMIF_T_CHAR (8−bit Signed Integer)

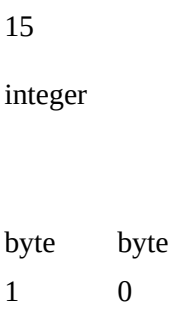
This stores an 8−bit signed integer

7
integer
byte 0

.

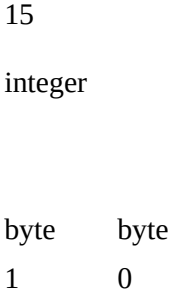
EMIF_T_USHORT (16−bit Unsigned Integer)

This stores a 16−bit unsigned integer, stored in Intel byte order. The least significant byte (byte 0) is stored first.



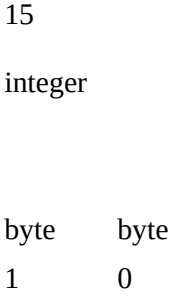
EMIF_T_SHORT (16−bit Signed Integer)

This stores a 16−bit two’s−complement integer, stored in Intel byte order. The least significant byte is stored first.



EMIF_T_ENUM (Enumerated Data Types)

This stores an enumerated data type as a 16−bit unsigned integer, stored in Intel byte order. The least significant byte is stored first. The list of strings associated with the type are defined in the data dictionary which is defined below. The first item in the list is indicated by 0.



EMIF_T_ULONG (32−bit Unsigned Integer)

This stores a 32−bit unsigned integer, stored in Intel byte order. The least significant byte is stored first.

31

integer

byte 3	byte 2	byte 1	byte 0

EMIF_T_LONG (32−bit Signed Integer)

This stores a 32−bit two’s−complement integer value, stored in Intel byte order. The least significant byte is stored first.

31

integer

byte 3	byte 2	byte 1	byte 0

EMIF_T_PTR (32−bit Unsigned Integer)

This stores a 32−bit unsigned integer, which is used to provide a byte address within the file. Byte 0 is the first byte, byte 1 is the second, etc. This allows for indexing into a 4−Gigabyte file, however most UNIX systems only allow 2−Gigabyte files.

i Currently, this element appears in the data dictionary as a EMIF_T_ULONG element. In future versions of the file format, the EMIF_T_PTR will be expanded to an 8−byte format which will allow indexing using 64 bits which allow addressing of 16 billion Gigabytes of file space.

31

integer

byte	byte	byte	byte
3	2	1	0

EMIF_T_TIME (32−bit Unsigned Integer)

This stores a 32−bit unsigned integer, which represents the number of seconds since 00:00:00 1 JAN 1970. This is the standard used in UNIX time keeping. The least significant byte is stored first.

31

integer

byte	byte	byte	byte
3	2	1	0

EMIF_T_FLOAT (Single Precision Floating Point)

Single precision floating point values are IEEE floating point values.

s = sign (0 = positive, 1 = negative)
exp = 8 bit excess 127 exponent
fraction = 24 bits of precision (includes 1 hidden bit)

31	30	22
s	exp	fraction

byte	byte	byte	byte
3	2	1	0

EMIF_T_DOUBLE (Double Precision Floating Point)

Double precision floating point data are IEEE double precision.

s = sign (0 = positive, 1 = negative)

exp = 11 bit excess 1023 exponent
fraction = 53 bits of precision (includes 1 hidden bit)

63	62	51
s	exp	fraction

byte 7	byte 6	byte 5	byte 4	byte 3	byte 2	byte 1	byte 0

EMIF_T_COMPLEX (Single Precision Complex)

A complex data element has a real part and an imaginary part. Single precision floating point values are IEEE floating point values.

s = sign (0 = positive, 1 = negative)
exp = 8 bit excess 127 exponent
fraction = 24 bits of precision (includes 1 hidden bit)

Real part: first single precision

31	30	22
s	exp	fraction

byte 3	byte 2	byte 1	byte 0

Imaginary part: second single precision

31	30	22
s	exp	fraction

byte 3	byte 2	byte 1	byte 0

EMIF_T_DCOMPLEX (Double Precision Complex)

A complex data element has a real part and an imaginary part. Double precision floating point data are IEEE double precision.

s = sign (0 = positive, 1 = negative)
exp = 11 bit excess 1023 exponent
fraction = 53 bits of precision (includes 1 hidden bit)

Real part: first double precision

63	62		51
s	exp		fraction

byte 7	byte 6	byte 5	byte 4	byte 3	byte 2	byte 1	byte 0

Imaginary part: second double precision

63	62		51
s	exp		fraction

byte 15	byte 14	byte 13	byte 12	byte 11	byte 10	byte 9	byte 8

EMIF_T_BASEDATA (Matrix of Numbers)

A BASEDATA is a generic two dimensional array of values. It can store any of the types of data used by IMAGINE. It is a variable length object whose size is determined by the data type, the number of rows, and the number of columns.

numrows: This indicates the number of rows of data in this item.

integer

byte	byte	byte	byte
3	2	1	0

numcolumns: This indicates the number of columns of data in this item.

31

integer

byte	byte	byte	byte
7	6	5	4

datatype: This indicates the type of data stored here. The types are:

	Data Type	**BytesPerObject**
0	EMIT_T_U1	1/8
1	EMIF_T_U2	1/4
3	EMIT_T_U4	1/2
4	EMIF_T_UCHAR	1
5	EMIF_T_CHAR	1
6	EMIF_T_USHORT	2
7	EMIF_T_SHORT	2
8	EMIF_T_ULONG	4
9	EMIF_T_LONG	4
10	EMIF_T_FLOAT	4
11	EMIF_T_DOUBLE	8
12	EMIF_T_COMPLEX	8
13	EMIF_T_DCOMPLEX	16

15

integer

byte 9 | byte 8

objecttype: This indicates the object type of the data. This is used in the IMAGINE Spatial Modeler. The valid values are:

0	SCALAR. This will not normally be the case, since a scalar has a single value.
1	TABLE: This indicates that the object is an array. The numcolumns should be 1.
2	MATRIX: This indicates the number of rows and columns is greater than one. This is used for Coefficient matrices, etc.
3	RASTER: This indicates that the number of rows and columns is greater than one and the data are just a part of a larger raster object. This would be the case for blocks of images which are written to the file.

15

integer

byte 11 | byte 10

data: This is the actual data. The number of bytes is given as:

bytecount = numrows * numcolumns * BytesPerObject

EMIF_M_INDIRECT (Indication of Indirect Data)

This is used when the following data belongs to an indirect reference of data. For example, when one object is defined by referring to another object.

The first four bytes provide the object repeat count.

31

integer

byte 3	byte 2	byte 1	byte 0

The next four bytes provide the file pointer which points to the data comprising the object.

31

integer

byte 7	byte 6	byte 5	byte 4

EMIF_M_PTR (Indication of Indirect Data)

This is used when the following data belong to an indirect reference of data of variable length. For example, when one object is defined by referring to another object. This is identical in file format to the EMIF_M_INDIRECT element. Its main difference is in the memory resident object which gets created. In the case of the EMIF_M_PTR the count and data pointer are placed into memory. Whereas only the data gets placed into memory when the EMIF_M_INDIRECT element is read in. (The size of the object is inherent in the data definitions.)

The first four bytes provide the object repeat count.

31

integer

byte	byte	byte	byte
3	2	1	0

The next four bytes provide the file pointer which points to the data comprising the object.

31

integer

byte	byte	byte	byte
7	6	5	4

MIF Data Dictionary

IMAGINE HFA files have a data dictionary that describes the contents of each of the different types of nodes. The dictionary is a compact ASCII string which is usually placed at the end of the file. A pointer to the start of the dictionary is stored in the header of the file.

Each object is defined like a structure in C, and consists of one or more items. Each item is composed of an ItemType and a name. The ItemType indicates the type of data and the name indicates the name by which the item will be known.

The syntax of the **`dictionary`** string is:

Dictionary	*ObjectDefinition[ObjectDefinition...]***.** The dictionary is one or more ObjectDefinitions terminated by a period. This is the complete collection of object type definitions.
ObjectDefinition	**{***ItemDefinition[ItemDefinition...]***}***name,* An ObjectDefinition is an ItemDefinition enclosed in braces {} followed by a name and terminated by a comma. This is a complete definition of a single object.

ItemDefinition	*number***:***[********|p***]ItemType[EnumData]name,*** An ItemDefinition is a number followed by a colon, followed optionally by either an asterisk or a p, followed by an ItemType, followed optionally by EnumData, followed by an item name, and terminated by a comma. This is the complete definition of a single Item. The * and the p both indicate that when the data are read into memory, they will not be placed directly into the structure being built, but that a new structure will be allocated and filled with the data. The pointer to that structure is placed into the initial structure. The asterisk indicates that the number of items in the indirect object is given by the number in the item definition. The p indicates that the number is variable. In both cases, the count precedes the data in the input stream.
EnumData	*number***:***name,[<name>,...]* EnumData is a number, followed by a colon, followed by one or more names each of which is terminated by a comma. The number defines the number of names which will follow. This is the complete set of names associated with an individual enum type.
name	Any sequence of alphanumeric characters excluding the comma.
number	A positive integer number. This composed of any sequence of these digits: 0,1,2,3,4,5,6,7,8,9.
ItemType	**1|2|4|c|C|s|S|l|L|f|d|t|m|M|b|e|o|x** This is used to indicate the type of an item. The following table indicates how the characters correspond to one of the basic EMIF_T types.

This table describes the single character codes used to identify the ItemType in the MIF Dictionary Definition.

The Interpretation column describes the type of data indicated by the item type. The Number of Bytes column is the number of bytes that the data type will occupy in the MIF file. If the number of bytes is not fixed, then it is given as dynamic.

ItemType	Interpretation	Number of Bytes
1	EMIF_T_U1	1
2	EMIF_T_U2	1
4	EMIF_T_U4	1
c	EMIF_T_UCHAR	1
C	EMIF_T_CHAR	1
e	EMIF_T_ENUM.	2
s	EMIF_T_USHORT	2
S	EMIF_T_SHORT	2
t	EMIF_T_TIME	4
l	EMIF_T_ULONG	4
L	EMIF_T_LONG	4
f	EMIF_T_FLOAT	4
d	EMIF_T_DOUBLE	8
m	EMIF_T_COMPLEX	8
M	EMIF_T_DCOMPLEX	16
b	EMIT_T_BASEDATA	dynamic
o	Previously defined object. This indicates that the description of the following data has been previously defined in the dictionary. This is like using a previously defined structure in a structure definition.	dynamic

x	Defined object for this entry. This indicates that the description of the following data follows. This is like using a structure definition within a structure definition.	dynamic

HFA Object Directory

The following section defines the list of objects which comprise ERDAS IMAGINE image files (.img extension). This is not a complete list because users and developers may create new items and add them to any ERDAS IMAGINE file.

The image files created and used by ERDAS IMAGINE are stored in a hierarchical file architecture (HFA). This format allows any number of different types of data elements to be stored in the file in a tree structured fashion. This tree is built of nodes which contain a variety of types of data. The contents of the nodes (as well as the structural information) is saved in the file in a machine independent format (MIF) which allows the files to be shared between computers of differing architectures.

Hierarchical File Architecture

The hierarchical file architecture maintains an object−oriented representation of data in an ERDAS IMAGINE disk file through use of a tree structure. Each object is called an entry and occupies one node in the tree. Each object has a name and a type. The type refers to a description of the data contained by that object. Additionally each object may contain a pointer to a subtree of more nodes. All entries are stored in MIF and can be accessed directly by name.

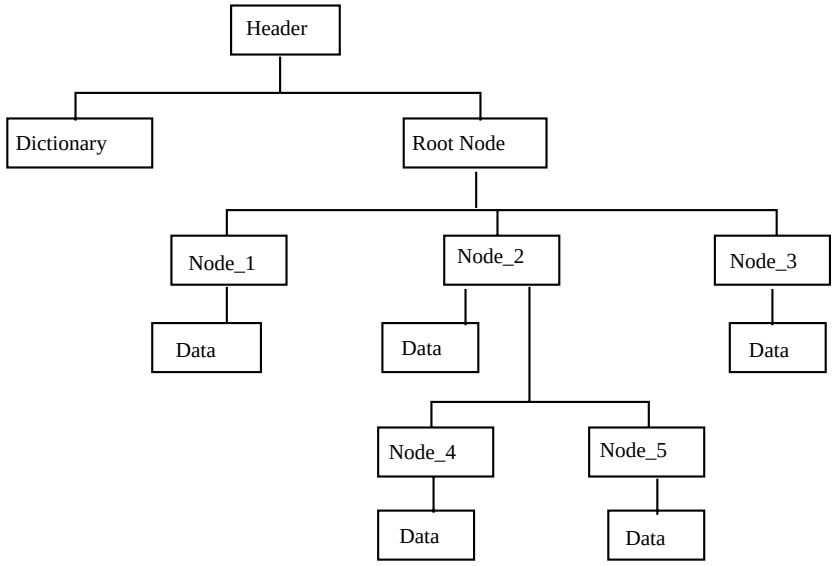


Figure 8: HFA File Structure

Nodes and Objects

Each node within the HFA tree structure contains an object and each object has its own data. The types of objects in a file are dependent upon the type of file. For example, a .img file will have different objects than an .ovr file because these files store different types of data. The list of objects in a file is not fixed. Objects may be added or removed depending on the data in the file (e.g., not every .img file with continuous raster layers will have a node for ground control points).

Figure 9, below, is an example of an HFA file structure for a thematic raster layer in a .img file. If there were more attributes in the ERDAS IMAGINE Raster Attribute Editor, then they would appear as objects under the Descriptor Table object.

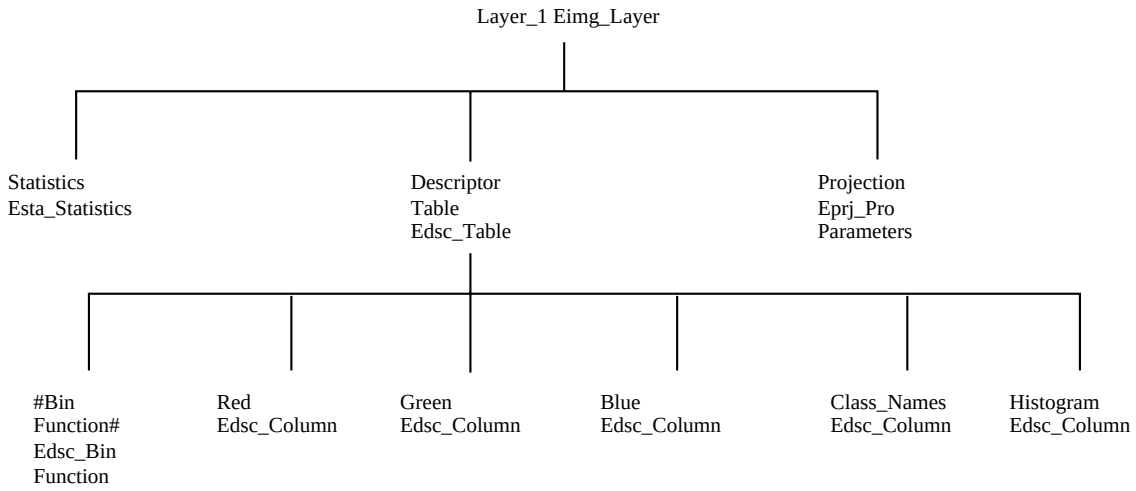


Figure 9: HFA File Structure Example

Pre−defined HFA File Object Types

There are three categories of pre−defined HFA File Object Types found in .img files:

- **Basic HFA File Object Types**
- **.img Object Types**
- **External File Format Header Object Types**

These sections list each object with two different detailed definitions. The first definition shows you how the object appears in the data dictionary in the HFA file. The second definition is a table that shows you the type, name, and description of each item in the object. An item within an object can be an element or another object.

i If an item is an element, then the item type is one of the basic types previously given with the EMIF_T_

prefix omitted. For example, the item type for EMIF_T_CHAR would be shown as CHAR.

If an item is a previously defined object type, then the type is simply the name of the previously defined item.

If the item is an array, then the number of elements is given in square brackets [n] after the type. For example, the type for an item with an array of 16 EMIF_T_CHAR would appear as CHAR[16]. If the item is an indirect item of fixed size (it is a pointer to an item), then the type is followed by an asterisk "*." For example, a pointer to an item with an array of 16 EMIF_T_CHAR would appear as CHAR[16] *. If the item is an indirect item of variable size (it is a pointer to an item and the number of items), then the type is followed by a "p." For example, a pointer to an item with a variable sized array of characters would look like CHAR p.

i If the item type is shown as PTR, then this item will be encoded in the data dictionary as a ULONG element.

Basic Objects of an HFA File

This is a list of types of basic objects found in all HFA files:

- **Ehfa_HeaderTag**
- **Ehfa_File**
- **Ehfa_Entry**

Ehfa_HeaderTag

The Ehfa_HeaderTag is used as a unique signature at the beginning of an ERDAS IMAGINE HFA file. It must always occupy the first 20 bytes of the file.

{16:clabel,1:lheaderPtr,}Ehfa_HeaderTag,

Type	Name	Description
CHAR[16]	label	This contains the string "EHFA_HEADER_TAG"
PTR	headerPtr	The file pointer to the Ehfa_File header record.

Ehfa_File

The Ehfa_File is composed of several main parts, including the free list, the dictionary, and the object tree. This

entry is used to keep track of these items in the file, since they may begin anywhere in the file.

{1:Lversion,1:lfreeList,1:lrootEntryPtr,1:SentryHeaderLength,1:ldictionaryPtr,}
Ehfa_File,

Type	Name	Description
LONG	version	This defines the version number of the ehfa file. It is currently 1.
PTR	freeList	This points to list of freed blocks within the file. This list is searched first whenever new space is needed. As blocks of space are released in the file, they are placed on the free list so that they may be reused later.
PTR	rootEntryPtr	This points to the root node of the object tree.
SHORT	entryHeaderLength	This defines the length of the entry portion of each node. Each node consists of two parts. The first part is the entry which contains the node name, node type, and parent/child information. The second part is the data for the node.
PTR	dictionaryPtr	This points to the starting position of the file for the MIF Dictionary. The dictionary must be read and decoded before any of the other objects in the file can be decoded.

Ehfa_Entry

The Ehfa_Entry contains the header information for each node in the object tree, including the name and type of the node as well as the parent/child information.

{1:lnext,1:lprev,1:lparent,1:lchild,1:ldata,1LdataSize,64:cname,32:ctype,
1:tmodTime,}Ehfa_Entry,

Type	Name	Description

PTR	next	This is a file pointer which gives the location of the next node in the tree at the current level. If this is the last node at this level, then this contains 0.
PTR	prev	This is a file pointer which gives the location of the previous node in the tree at the current level. If this is the first node at this level, then this contains 0.
PTR	parent	This is a file pointer which gives the location of the parent for this node. This is 0 for the root node.
PTR	child	This is a file pointer which gives the location of the first of the list of children for this node. If there are no children, then this contains 0.
PTR	data	This points to the data for this node. If there is no data for this node then it contains 0.
LONG	dataSize	This contains the number of bytes contained in the data record associated with this node.
CHAR[64]	name	This contains a NULL terminated string that is the name for this node. The string can be no longer then 64 bytes including the NULL terminator byte.
CHAR[32]	type	This contains a NULL terminated string which names the type of data to be found at this node. The type must match one of the types found in the data dictionary. The type name can be no longer then 32 bytes including the NULL terminator byte.
TIME	modTime	This contains the time of the last modification to the data in this node.

Objects of a .img File

This is a list of types of pre−defined objects commonly found in .img HFA files:

- **Eimg_Layer**
- **Eimg_Layer_SubSample**
- **Eimg_NonInitializedValue**
- **Ehfa_Layer**
- **Edms_VirtualBlockInfo**
- **Edms_FreeIDList**
- **Edms_State**
- **Edsc_Table**
- **Edsc_BinFunction**
- **Edsc_Column**
- **Eded_ColumnAttributes_1**
- **Esta_Statistics**
- **Esta_Covariance**
- **Esta_SkipFactors**
- **Esta_ExcludedValues**
- **Eprj_Datum**
- **Eprj_Spheroid**
- **Eprj_ProParameters**
- **Eprj_Coordinate**
- **Eprj_Size**
- **Eprj_MapInfo**
- **Efga_Polynomial**
- **Calibration_Node**

Eimg_Layer

An Eimg_Layer object is the base node for a single layer of imagery. This object describes the basic information for the layer, including its width and height in pixels, its data type, and the width and height of the blocks used to store the image. Other information such as the actual pixel data, map information, projection information, etc., are stored as child objects under this node. The child objects that are usually found under the Eimg_Layer include:

- RasterDMS (an Edms_State which actually contains the imagery)
- Descriptor_Table (an Edsc_Table object which contains the histogram and other pixel value related data)
- Projection (an Eprj_ProParameters object which contains the projection information)
- Map_Info (an Eprj_MapInfo object which contains the map information)
- Ehfa_Layer (an Ehfa_Layer object which describes the type of data in the layer)

{1:lwidth,1:lheight,1:e3:thematic,athematic,fft of real valued data,
layerType,1e13:u1,u2,u4,u8, s8,u16,s16,u32,s32,f32,f64,c64,c128,pixelType,
1:lblockWidth,1:lblockHeight,} Eimg_Layer,

Type	Name	Description
LONG	width	The width of the layer in pixels.
LONG	height	The height of the layer in pixels.
ENUM	layerType	The type of layer. 0="thematic" 1="athematic"
ENUM	pixelType	The type of the pixels. 0="u1" 1="u2" 2="u4" 3="u8" 4="s8" 5="u16" 6="s16" 7="u32" 8= "s32" 9="f32" 10="f64" 11="c64"

LONG	blockWidth	The width of each block in the layer.
LONG	blockHeight	The height of each block in the layer.

Eimg_Layer_SubSample

An Eimg_Layer_SubSample object is a node which contains a subsampled version of the layer defined by the parent node. The node of this form are named _ss_2, _ss_4, _ss_8, etc. This stands for SubSampled by 2, SubSampled by 4, etc. This node will have an Edms_State node called RasterDMS and an Ehfa_Layer node called Ehfa_layer under it. This will be present if pyramid layers have been computed.

{1:lwidth,1:lheight,1:e3:thematic,athematic,fft of real valued data,
layerType,1e13:u1,u2,u4,u8, s8,u16,s16,u32,s32,f32,f64,c64,c128,pixelType,
1:lblockWidth,1:lblockHeight,} Eimg_Layer_SubSample,

Type	**Name**	**Description**
LONG	width	The width of the layer in pixels.
LONG	height	The height of the layer in pixels.
ENUM	layerType	The type of layer. 0 ="thematic" 1 ="athematic"
ENUM	pixelType	The type of the pixels. 0="u1" 1="u2" 2="u4" 3="u8" 4="s8" 5="u16" 6="s16" 7="u32" 8="s32" 9="f32" 10="f64" 11="c64" 12="c128"

LONG	blockWidth	The width of each block in the layer.
LONG	blockHeight	The height of each block in the layer.

Eimg_NonInitializedValue

The Eimg_NonInitializedValue object is used to record the value that is to be assigned to any uninitialized blocks of raster data in a layer.

{1:*bvalueBD,}Eimg_NonInitializedValue,

Type	Name	Description
BASEDATA *	valueBD	A basedata structure containing the excluded values

Ehfa_Layer

The Ehfa_Layer is used to indicate the type of layer. The initial design for the IMAGINE files allowed for both raster and vector layers. Currently, the vector layers have not been implemented.

{1:e2:raster,vector,type,1:ldictionaryPtr,}Ehfa_Layer,

Type	Name	Description
ENUM	type	The type of layer. 0="raster" 1="vector"
ULONG	dictionaryPtr	This points to a dictionary entry which describes the data. In the case of raster data, it points to a dictionary pointer which describes the contents of each block via the **RasterDMS** definition given below.

RasterDMS

The RasterDMS object definition must be present in the EMIF dictionary pointed to by an Ehfa_Layer object that is of type "raster". It describes the logical make−up of a block of raster data in the Ehfa_Layer. The physical representation of the raster data is actually managed by the DMS system through objects of type Ehfa_Layer and Edms_State. The RasterDMS definition should describe the raster data in terms of total number of data values in a block and the type of data value.

{<n>:<t>data,}RasterDMS,

Type	Name	Description
<t>[<n>]	data	The data is described in terms of total number, <n>, of data file values in a block of the raster layer (which is simply <block width> * <block height>) and the data value type, <t>, which can have any one of the following values: 1 − Unsigned 1−bit 2 − Unsigned 2−bit 4 − Unsigned 4−bit c − Unsigned 8−bit C − Signed 8−bit s − Unsigned 16−bit S − Signed 16−bit l − Unsigned 32−bit L − Signed 32−bit f − Single precision floating point d − Double precision floating point m − Single precision complex M − Double precision complex

Edms_VirtualBlockInfo

An Edms_VirtualBlockInfo object describes a single raster data block of a layer. It describes where to find the

data in the file, how many bytes are in the data block, and how to unpack the data from the block. For uncompressed data the unpacking is straight forward. The scheme for compressed data is described below.

{1:SfileCode,1:loffset,1:lsize,1:e2:false,true,logvalid,1:e2:no compression,ESRI GRID compression,compressionType,1:LblockHeight,}Edms_VirtualBlockInfo,

Type	Name	Description
SHORT	fileCode	This is included to allow expansion of the layer into multiple files. The number indicates the file in which the block is located. Currently this is always 0, since the multiple file scheme has not been implemented.
PTR	offset	This points to the byte location in the file where the block data actually resides.
LONG	size	The number of bytes in the block.
ENUM	logvalid	This indicates whether the block actually contains valid data. This allows blocks to exist in the map, but not in the file. 0="false" 1="true"
ENUM	compressionType	This indicates the type of compression used for this block. 0="no compression" 1="ESRI GRID compression" No compression indicates that the data located at offset are uncompressed data. The stream of bytes is to be interpreted as a sequence of bytes which defines the data as indicated by the data type. The ESRI GRID compression is a two stage run−length encoding.

For uncompressed blocks, the data are simply packed into the block one pixel value at a time. Each pixel is read from the block as indicated by its data type. All non−integer data are uncompressed.

The compression scheme used by ERDAS IMAGINE is a two level run−length encoding scheme. If the data are

an integral type, then the following steps are performed:

- The minimum and maximum values for a block are determined.
- The byte size of the output pixels is determined by examining the difference between the maximum and the minium. If the difference is less than or equal to 256, then 8−bit data are used. If the difference is less than 65,536 then, 16−bit data are used, otherwise 32−bit data are used.
- The minimum is subtracted from each of the values.
- A run−length encoding scheme is used to encode runs of the same pixel value. The data minimum value occupies the first 4 bytes of the block. The number of run−length segments occupies the next 4 bytes, and the next 4 bytes are an offset into the block which indicates where the compressed pixel values begin. The next byte indicates the number of bits per pixel (1,2,4,8,16,32). These four values are encoded in the standard MIF format (ULONG). Following this is the list of segment counts, following the segment counts are the pixel values. There is one segment count per pixel value.

i No compression scheme is used if the data are non−integral.

min	num segments	data offset	numbits per value	data counts	data values

Each data count is encoded as follows:

next 8 bits	next 8 bits	next 8 bits	byte count	high 6 bits

byte 3	byte 2	byte 1	byte 0

There may be 1, 2, 3, or 4 bytes per count. The first two bits of the first count byte contains 0,1,2,3 indicating that the count is contained in 1, 2,3, or 4 bytes. Then the rest of the byte (6 bits) represent the six most significant bytes of the count. The next byte, if present, represents decreasing significance.

i This order is different than the rest of the package. This was done so that the high byte with the encoded byte count would be first in the byte stream. This pattern is repeated as many times as indicated by the numsegments field.

The data values are compressed into the remaining space packed into as many bits per pixel as indicated by the

numbitpervalue field.

Edms_FreeIDList

An Edms_FreeIDList is used to track blocks which have been freed from the layer. The freelist consists of an array of min/max pairs which indicate unused contiguous blocks of data which lie within the allocated layer space. Currently this object is unused and reserved for future expansion.

{1:Lmin,1:Lmax,}Edms_FreeIDList,

Type	Name	Description
LONG	min	The minimum block number in the group.
LONG	max	The maximum block number in the group.

Edms_State

The Edms_State describes the location of each of the blocks of a single layer of imagery. Basically, this object is an index of all of the blocks in the layer.

{1:lnumvirtualblocks,1:lnumobjectsperblock,1:lnextobjectnum,
1:e2:no compression,RLC compression,compressionType,
0:poEdms_VirtualBlockInfo,blockinfo,0:poEdms_FreeIDList,freelist,
1:tmodTime,}Edms_State,

Type	Name	Description
LONG	numvirtual blocks	The number of blocks in this layer.
LONG	numobjectsper block	The number of pixels represented by one block.
LONG	nextobjectnum	Currently, this type is not being used and is reserved for future expansion.

ENUM	compressionType	This indicates the type of compression used for this block. 0="no compression" 1="ESRI GRID compression" No compression indicates that the data located at offset are uncompressed data. The stream of bytes is to be interpreted as a sequence of bytes which defines the data as indicated by the data type. The ESRI GRID compression is a two stage run−length encoding.
Edms_VirtualBolckInfo p	blockinfo	This is the table of entries which describes the state and location of each block in the layer.
Edms_FreeIDList p	freelist	Currently, this type is not being used and is reserved for future expansion.
TIME	modTime	This is the time of the last modification to this layer.

Edsc_Table

An Edsc_Table is a base node used to store columns of information. This serves simply as a parent node for each of the columns which are a part of the table.

{1:lnumRows,} Edsc_Table,

Type	Name	Description
LONG	numRows	This defines the number of rows in the table.

Edsc_BinFunction

The Edsc_BinFunction describes how pixel values from the associated layer are to be mapped into an index for the columns.

{1:lnumBins,1:e4:direct,linear,logarithmic,explicit,binFunction Type,
1:dminLimit,1:dmaxLimit,1:*bbinLimits,} Edsc_BinFunction,

Type	Name	Description
LONG	numBins	The number of bins.
ENUM	binFunction Type	The type of bin function. 0="direct" 1="linear" 2="exponential" 3="explicit"
DOUBLE	minLimit	The lowest value defined by the bin function.
DOUBLE	maxLimit	The highest value defined by the bin function.
BASEDATA	binLimits	The limits used to define the bins.

The following table describes how the binning functions are used.

Bin Type	Description
DIRECT	Direct binning means that the pixel value minus the minimum is used as is with no translation to index into the columns. For example, if the minimum value is zero, then value 0 is indexed into location 0, 1 is indexed into 1, etc.
LINEAR	Linear binning means that the pixel value is first scaled by the formula: index = (value−minLimit)*numBins/(maxLimit−minLimit). This allows a very large range of data, or even floating point data, to be used to index into a table.
EXPONENTIAL	Exponential binning is used to compress data with a large dynamic range. The formula used is index = numBins*(log(1+(value−minLimit)) / (maxLimit−minLimit).

EXPLICIT	Explicit binning is used to map the data into indices using an arbitrary set of boundaries. The data are compared against the limits set in the binLimit table. If the pixel is less than or equal to the first value, then the index is 0. If the pixel is less than or equal to the next value, then the index is 1, etc.

Edsc_Column

The columns of information which are stored in a table are stored in this format.

{1:lnumRows,1:LcolumnDataPtr,1:e4:integer,real,comples,string,dataType,
1:lmaxNumChar,} Edsc_Column,

Type	Name	Description
LONG	numRows	The number of rows in this column.
PTR	columnDataPtr	Starting point of column data in the file. This points to the location in the file which contains the data.
ENUM	dataType	The data type of this column 0="integer" (EMIF_T_LONG) 1="real" (EMIF_T_DOUBLE) 2="complex" (EMIF_T_DCOMPLEX) 3="string" (EMIF_T_CHAR)
LONG	maxNumChars	The maximum string length (for string data only). It is 0 if the type is not a String.

The types of information stored in columns are given in the following table.

Name	Data Type	Description
Histogram	real	This is found in the descriptor table of almost every layer. It defines the number of pixels which fall into each bin.

Class_Names	string	This is found in the descriptor table of almost every thematic layer. It defines the name for each class.
Red	real	This is found in the descriptor table of almost every thematic layer. It defines the red component of the color for each class. The range of the value is from 0.0 to 1.0.
Green	real	This is found in the descriptor table of almost every thematic layer. It defines the green component of the color for each class. The range of the value is from 0.0 to 1.0.
Blue	real	This is found in the descriptor table of almost every thematic layer. It defines the blue component of the color for each class. The range of the value is from 0.0 to 1.0.
Opacity	real	This is found in the descriptor table of almost every thematic layer. It defines the opacity associated with the class. A value of 0 means that the color will be solid. A value of 0.5 mean that 50% of the underlying pixel would show through, and 1.0 means that all of the pixel value in the underlying layer would show through.
Contrast	real	This is found in the descriptor table of most continuous raster layers. It is used to define an intensity stretch which is normally used to improve contrast. The table is stored as normalized values from 0.0 to 1.0.
GCP_Names	string	This is found in the GCP_Table in files which have ground control points. This is the table of names for the points.
GCP_xCoords	real	This is found in the GCP_Table in files which have ground control points. This is the X coordinate for the point.

GCP_yCoords	real	This is found in the GCP_Table in files which have ground control points. This is the Y coordinate for the point.
GCP_Color	string	This is found in the GCP_Table in files which have ground control points. This is the name of the color that is used to display this point.

Eded_ColumnAttributes_1

The Eded_ColumnAttributes_1 stores the descriptor column properties which are used by the Raster Attribute Editor for the format and layout of the descriptor column display in the Raster Attribute Editor CellArray. The properties include the position of the descriptor column within the CellArray, the name, alignment, format, and width of the column, whether the column is editable, the formula (if any) for the column, the units (for numeric data), and whether the column is a component of a color column. Each Eded_ColumnAttributes_1 is a child of the Edsc_Column containing the data for the descriptor column. The properties for a color column are stored as a child of the Eded_ColumnAttributes_1 for the red component of the color column.

{1:lposition,0:pcname,1:e2:FALSE,TRUE,editable,
1:e3:LEFT,CENTER,RIGHT,alignment,0:pcformat,
1:e3:DEFAULT,APPLY,AUTO−APPLY,formulamode,0:pcformula,1:dcolumnwidth,
0:pcunits,1:e5:NO_COLOR,RED,GREEN,BLUE,COLOR,colorflag,0:pcgreenname,
0:pcbluename,}Eded_ColumnAttributes_1,

Type	**Name**	**Description**
LONG	position	The position of this descriptor column in the Raster Attribute Editor CellArray. The positions for all descriptor columns are sorted and the columns are displayed in ascending order.
CHAR P	name	The name of the descriptor column. This is the same as the name of the parent Edsc_Column node, for all columns except color columns, Color columns have no corresponding Edsc_Column.

ENUM	editable	Specifies whether this column is editable. 0 = NO 1 = YES
ENUM	alignment	Alignment of this column in CellArray. 0 = LEFT 1 = CENTER 2 = RIGHT
CHAR P	format	The format for display of numeric data.
ENUM	formulamode	Mode for formula application. 0 = DEFAULT 1 = APPLY 2 = AUTO−APPLY
CHAR P	formula	The formula for the column.
DOUBLE	columnwidth	The width of the CellArray column
CHAR P	units	The name of the units for numeric data stored in the column.
ENUM	colorflag	Indicates whether column is a color column, a component of a color column, or a normal column. 0 = NO_COLOR 1 = RED 2 = GREEN 3 = BLUE 4 = COLOR
CHAR P	greenname	Name of green component column associated with color column. Empty string for other column types.
CHAR P	bluename	Name of blue component column associated with color column. Empty string for other column types.

Esta_Statistics

The Esta_Statistics is used to describe the statistics for a layer.

{1:dminimum,1:dmaximum,1:dmean,1:dmedian,1d:mode,1:dstddev,}
Esta_Statistics,

Type	Name	Description
DOUBLE	minimum	The minimum of all of the pixels in the image. This may exclude values as defined by the user.
DOUBLE	maximum	The maximum of all of the pixels in the image. This may exclude values as defined by the user.
DOUBLE	mean	The mean of all of the pixels in the image. This may exclude values as defined by the user.
DOUBLE	median	The median of all of the pixels in the image. This may exclude values as defined by the user.
DOUBLE	mode	The mode of all of the pixels in the image. This may exclude values as defined by the user.
DOUBLE	stddev	The standard deviation of the pixels in the image. This may exclude values as defined by the user.

Esta_Covariance

The Esta_Covariance object is used to record the covariance matrix for the layers in a .img file

{1:bcovariance,}Esta_Covariance,

Type	Name	Description
BASEDATA	covariance	A basedata structure containing the covariance matrix

Esta_SkipFactors

The Esta_SkipFactors object is used to record the skip factors that were used when the statistics or histogram was

calculated for a raster layer or when the covariance was calculated for a .img file.

{1:LskipFactorX,1:LskipFactorY,}Esta_SkipFactors,

Type	Name	Description
LONG	skipFactorX	The horizontal sampling interval used for statistics measured in image columns/sample
LONG	skipFactorY	The vertical sampling interval used for statistics measured in image rows/sample

Esta_ExcludedValues

The Esta_ExcludedValues object is used to record the values that were excluded from consideration when the statistics or histogram was calculated for a raster layer or when the covariance was calculated for a .img file.

{1:*bvalueBD,}Esta_ExcludedValues,

Type	Name	Description
BASEDATA *	valueBD	A basedata structure containing the excluded values

Eprj_Datum

The Eprj_Datum object is used to record the datum information which is part of the projection information for a .img file.

{0:pcdatumname,1:e3:EPRJ_DATUM_PARAMETRIC,EPRJ_DATUM_GRID,
EPRJ_DATUM_REGRESSION,type,0:pdparams,0:pcgridname,}Eprj_Datum,

Type	Name	Description
CHAR	datumname	The datum name.
ENUM	type	The datum type which could be one of three different types: parametric type, grid type and regression type.

DOUBLE	params	The seven parameters of a parametric datum which describe the translations, rotations and scale change between the current datum and the reference datum WGS84.
CHAR	gridname	The name of a grid datum file which stores the coordinate shifts among North America Datums NAD27, NAD83 and HARN.

Eprj_Spheroid

The Eprj_Spheroid is used to describe spheroid parameters used to describe the shape of the earth.

{0:pcsphereName,1:da,1:db,1:deSquared,1:dradius,}Eprj_Spheroid,

Type	Name	Description
CHAR p	sphereName	The name of the spheroid/ellipsoid. This name is can be found in: <IMAGINE_HOME>/etc/spheroid.tab.
DOUBLE	a	The semi−major axis of the ellipsoid in meters.
DOUBLE	b	The semi−minor axis of the ellipsoid in meters.
DOUBLE	eSquared	The eccentricity of the ellipsoid, squared.
DOUBLE	radius	The radius of the spheroid in meters.

Eprj_ProParameters

The Eprj_Parameters is used to define the map projection for a layer.

{1:e2:EPRJ_INTERNAL,EPRJ_EXTERNAL,proType,1:lproNumber,
0:pcproExeName,0:pcproName,1:lproZone,0:pdproParams,
1:*oEprj_Spheroid,proSpheroid,}Eprj_ProParameters,

Type	Name	Description

ENUM	proType	This defines whether the projection is internal or external. 0="EPRJ_INTERNAL" 1=" EPRJ_EXTERNAL"
LONG	proNumber	The projection number for internal projections. The current internal projections are: 0="Geographic(Latitude/Longitude)" 1="UTM" 2="State Plane" 3="Albers Conical Equal Area" 4="Lambert Conformal Conic" 5="Mercator" 6="Polar Stereographic" 7="Polyconic" 8="Equidistant Conic" 9="Transverse Mercator" 10="Stereographic" 11="Lambert Azimuthal Equal−area" 12="Azimuthal Equidistant" 13="Gnomonic" 14="Orthographic" 15="General Vertical Near−Side Perspective" 16="Sinusoidal" 17="Equirectangular" 18="Miller Cylindrical" 19="Van der Grinten I" 20="Oblique Mercator (Hotine)" 21="Space Oblique Mercator" 22="Modified Transverse Mercator"
CHAR p	proExeName	The name of the executable to run for an external projection.
CHAR p	proName	The name of the projection. This will be one of the names given above in the description of proNumber.

LONG	proZone	The zone number for internal State Plane or UTM projections.
DOUBLE p	proParams	The array of parameters for the projection.
Eprj_Spheroid *	proSpheroid	The parameters of the spheroid used to approximate the earth. See the description for the Eprj_Spheroid object, above.

The following table defines the contents of the proParams array which is defined above. The Parameters column defines the meaning of the various elements of the proParams array for the different projections. Each one is described by one or more statements of the form n: Description. n is the index into the array.

	Name	**Parameters**
0	"Geographic(Latitude/Longitude)"	None Used
1	"UTM"	3: 1=North, −1=South
2	"State Plane"	0: 0=NAD27, 1=NAD83
3	"Albers Conical Equal Area"	2: Latitude of 1st standard parallel 3: Latitude of 2nd standard parallel 4: Longitude of central meridian 5: Latitude of origin of projection 6: False Easting 7: False Northing
4	"Lambert Conformal Conic"	2: Latitude of 1st standard parallel 3: Latitude of 2nd standard parallel 4: Longitude of central meridian 5: Latitude of origin of projection 6: False Easting 7: False Northing
5	"Mercator"	4: Longitude of central meridian 5: Latitude of origin of projection 6: False Easting 7: False Northing

6	"Polar Stereographic"	4: Longitude directed straight down below pole of map. 5: Latitude of true scale. 6: False Easting 7: False Northing.
7	"Polyconic"	4: Longitude of central meridian 5: Latitude of origin of projection 6: False Easting 7: False Northing
8	"Equidistant Conic"	2: Latitude of standard parallel (Case 0) 2: Latitude of 1st Standard Parallel (Case 1) 3: Latitude of 2nd standard Parallel (Case 1) 4: Longitude of central meridian 5: Latitude of origin of projection 6: False Easting 7: False Northing 8: 0=Case 0, 1=Case 1.
9	"Transverse Mercator"	2: Scale Factor at Central Meridian 4: Longitude of center of projection 5: Latitude of center of projection 6: False Easting 7: False Northing
10	"Stereographic"	4: Longitude of center of projection 5: Latitude of center of projection 6: False Easting 7: False Northing
11	"Lambert Azimuthal Equal−area"	4: Longitude of center of projection 5: Latitude of center of projection 6: False Easting 7: False Northing
12	"Azimuthal Equidistant"	4: Longitude of center of projection 5: Latitude of center of projection 6: False Easting 7: False Northing

13	"Gnomonic"	4: Longitude of center of projection 5: Latitude of center of projection 6: False Easting 7: False Northing
14	"Orthographic"	4: Longitude of center of projection 5: Latitude of center of projection 6: False Easting 7: False Northing
15	"General Vertical Near−Side Perspective	2: Height of perspective point above sphere. 4: Longitude of center of projection 5: Latitude of center of projection 6: False Easting 7: False Northing
16	"Sinusoidal"	4: Longitude of central meridian 6: False Easting 7: False Northing
17	"Equirectangular"	4: Longitude of central meridian 5: Latitude of True Scale. 6: False Easting 7: False Northing
18	"Miller Cylindrical"	4: Longitude of central meridian 6: False Easting 7: False Northing
19	"Van der Grinten I"	4: Longitude of central meridian 6: False Easting 7: False Northing

20	"Oblique Mercator (Hotine)"	2: Scale Factor at center of projection 3: Azimuth east of north for central line. (Case 1) 4: Longitude of point of origin (Case 1) 5: Latitude of point of origin. 6: False Easting 7: False Northing. 8: Longitude of 1st Point defining central line (Case 0) 9: Latitude of 1st Point defining central line (Case 0) 10: Longitude of 2nd Point defining central line. (Case 0) 11: Latitude of 2nd Point defining central line (Case 0). 12: 0=Case 0, 1=Case 1
21	"Space Oblique Mercator"	4: Landsat Vehicle ID (1−5) 5: Orbital Path Number (1−251 or 1−233) 6: False Easting 7: False Northing
22	"Modified Transverse Mercator"	6: False Easting 7: False Northing

Eprj_Coordinate

An Eprj_Coordiante is a pair of doubles used to define X and Y.

{1:dx,1:dy,}Eprj_Coordinate,

Type	Name	Description
DOUBLE	x	The X value of the coordinate.
DOUBLE	y	The Y value of the coordinate.

Eprj_Size

The Eprj_Size is a pair of doubles used to define a rectangular size.

{1:dx,1:dy,}Eprj_Size,

Type	Name	Description
DOUBLE	width	The X value of the coordinate.
DOUBLE	height	The Y value of the coordinate.

Eprj_MapInfo

The Eprj_MapInfo object is used to define the basic map information for a layer. It defines the map coordinates for the center of the upper left and lower right pixels, as well as the cell size and the name of the map projection.

{0:pcproName,1:*oEprj_Coordinate,upperLeftCenter,
1:*oEprj_Coordinate,lowerRightCenter,1:*oEprj_Size,pixelSize,
0:pcunits,}Eprj_MapInfo,

Type	Name	Description
CHAR p	proName	The name of the projection.
Eprj_ Coordinate *	upperLeftCenter	The coordinates of the center of the upper left pixel.
Eprj_ Coordinate *	lowerRightCenter	The coordinates of the center of the lower right pixel.
Eprj_Size *	pixelSize	The size of the pixel in the image.
CHAR *	units	The units of the above values.

Efga_Polynomial

The Efga_Polynomial is used to store transformation coefficients created by the IMAGINE GCP Editor.

{1:Lorder,1:Lnumdimtransforms,1:numdimpolynomial,1:Ltermcount,
1:*exponentList,1:bpolycoefmtx,1:bpolycoefvector,}Efga_Polynomial,

Type	Name	Description

LONG	order	The order of the polynomial.
LONG	numdimtransform	The number of dimensions of the transformation (always 2).
LONG	numdimpolynomial	The number of dimensions of the polynomial (always 2).
LONG	termcount	The number of terms in the polynomial.
LONG *	exponentlist	The ordered list of powers for the polynomial.
BASEDATA	polycoefmtx	The polynomial coefficients.
BASEDATA	polycoefvector	The polynomial vectors.

Calibration_Node

An object of type Calibration_Node is an empty object — it contains no data. A node of this type simply serves as the parent node of four related child objects. The children of the Calibration_Node are used to provide information which converts pixel coordinates to map coordinates and vice versa.There is no dictionary definition for this object type. A node of this type will be a child of the root node and will be named "Calibration." The "Calibration" node will have the four children described below.

Node	Object Type	Description
Projection	Eprj_ProParameters	The projection associated with the output coordinate system.
Map_Info	Eprj_MapInfo	The nominal map information associated with the transformation.
InversePolynomial	Efga_Polynomial	This is the nth order polynomial coefficient used to convert from map coordinates to pixel coordinates.
ForwardPolynomial	Efga_Polynomial	This is the nth order polynomial used to convert from pixel coordinates to map coordinates

External File Format Header Object Tpyes

The following sections list the types of objects found in .img HFA files that have been imported from exteral data files:

- **ADRG Header**
- **ADRI Header**
- **AVHRR Header**
- **DEM Header**
- **DOQ Header**
- **DTED Header**
- **MSS Header**
- **SPOT Header**
- **TM Header**

ADRG Header

The following table defines the contents of the record written to an IMAGINE .img file which has been read from an ADRG source. The fields are copied from the Volume Header file ("TRANSH01.THF") and from the individual Distribution Rectangle (DR) headers. If the value is a string it is left unchanged. If the value is a number it is converted from ASCII to binary. All strings are NULL terminated. The description column indicates the source of the information. The three codes in the Description column represent the file name, field name, and subfield names. The filename codes are THF for Transmittal Header File and GEN for General Information File. For example, the description "THF, VDR, URF" means the URF subfield within the VDR field within the Transmittal Header File. It is assumed that the user has access to the ADRG documents.

.

Type	Name	Description
CHAR [Var. Length]	Originator	THF, VDR, VOO Name & address of originator

CHAR[17]	StockNum	THF, VDR, URF DMA stock number
SHORT	VolEdNum	THF, VDR, EDN Volume edition number
TIME	PubDate	THF, VDR, DAT Publication date
CHAR	SecurityClassification	THF, QSR, QSS Security classification
ENUM	AgencyDeterminationReqd	THF, QSR, QOD Originating agency's determination required NO or YES
TIME	DowngradeDate	THF, QSR, DAT Date of downgrading
CHAR [Var. Length]	Releasability	THF, QSR, QLE Releasability statement
CHAR [Var. Length]	SpecIdent	THF, QUV, SRC Specification identification for ADRG
TIME	SpecDate	THF, QUV, DAT Specification date
CHAR[21]	SpecAmendmentNum	THF, QUV, SPA ADRG Specification amendment number
CHAR[9]	DistRectName	THF, FDR, NAM Name of the DR directory
CHAR[13]	ImageName	GEN (OVERVIEW_RECORD),SPR,BAD or GEN (GENERAL_INFORMATION_RECORD), SPR, BAD or SOU,SPR,BAD Name of the image file

ENUM	ImageType	Type of image UNKNOWN, ZDR, LEGEND or OVERVIEW
SHORT	Zone	GEN, GEN, ZNA ARC Zone number

ADRI Header

The following table defines the contents of the record written to an IMAGINE .img file which has been read from an ADRI source. The fields are copied from the Volume Header file and from the individual Distribution Rectangle (DR) headers. If the value is a string it is left unchanged. If the value is a number it is converted from ASCII to binary. All strings are NULL terminated. The description column indicates the source of the information. The three codes in the Description column represent the file name, field name, and subfield names. The file name codes are THF for Transmittal Header File, GEN for General Information File, and QAL for Quality File. For example, the description "QAL, QUP, SPA" means the SPA subfield within the QUP (Quality Up To Dateness) field within the Quality File. Additional comments are added when necessary to avoid ambiguity. It is assumed that the user has access to the ADRI documentation.

.

Type	**Name**	**Description**
CHAR[Var. Length]	Originator	THF, VDR, VOO
CHAR[17]	StockNum	THF, VDR, URF
SHORT	VolEdNum	THF, VDR, EDN
TIME	PubDate	THF, VDR, DAT
CHAR	SecurityClassification	THF, QSR, QSS
ENUM	AgencyDeterminationReqd	THF, QSR, QOD NO or YES
TIME	DowngradeDate	THF, QSR, DAT
CHAR[Var. Length]	Releasability	THF, QSR, QLE
CHAR[Var. Length]	SpecIdent	THF, QUV, SRC
TIME	SpecDate	THF, QUV, DAT

CHAR[21]	SpecAmendmentNum	THF, QUV, SPA
ENUM	ImageType	OVERVIEW or ZDR
CHAR[9]	DistRectName	Name of the DR Directory
CHAR[13]	ImageName	Name of the Image File
SHORT	DataStructureType	GEN, OVI, STR or GEN, GEN, STR
DOUBLE	DataDensityEW	GEN, GEN, LOD
DOUBLE	DataDensityNS	GEN, GEN, LAD
SHORT	DataDensityUnit	GEN, GEN, UNI
DOUBLE	Ulx	GEN, GEN, NWO
DOUBLE	Uly	GEN, GEN, NWA
DOUBLE	Lrx	GEN, GEN, SEO
DOUBLE	Lry	GEN, GEN, SEA
DOUBLE	Scale	GEN, GEN, SCA
SHORT	Zone	GEN, GEN, ZNA
DOUBLE	GSD	GEN, GEN, PSP
ENUM	Rectified	GEN, GEN, IMR NO or YES
DOUBLE	Asz	GEN, GEN, ARV
DOUBLE	Bs	GEN, GEN, BRV
CHAR[Var. Length]	Text	GEN, GEN, TXT
struct BandInfo [Var. Length]	BandInfo	GEN, BDF (See BandInfo Structure table below)
CHAR[21]	EditionNumber	QAL, QUP, EDN
TIME	CreationDate	QAL, QUP, DAT (Creation date of ZDR)

TIME	RevisionDate	QAL, QUP, DAT (Date of revision/update)
SHORT	RecompilationCount	QAL, QUP, REC
SHORT	RevisionCount	QAL, QUP, REV
CHAR[Var. Length]	ADRIProduct	QAL, QUP, SRC
TIME	ProductSpecDate	QAL, QUP, DAT (ADRI product specification date)
CHAR[21]	ProductSpecNum	QAL, QUP, SPA
TIME	EarliestDate	QAL, QUP, DAT (Earliest date of source image in DR)
TIME	LatestDate	QAL, QUP, DAT (Latest date of source image in DR)

Type	**Name**	**Description**
CHAR[6]	BandColor	GEN, BDF, BID
LONG	LowerEdgeWavelength	GEN, BDF, WS1
LONG	UpperEdgeWavelength	GEN, BDF, WS2

BandInfo Structure

AVHRR Header

The following table defines the contents of the record written to an IMAGINE .img file which has been read from an AVHRR source. The fields are copied from the TBM and Data Set Headers. Note that not all AVHRR files contain a TBM header, so the TBM fields may be empty or zero. TBM header fields are indicated in the Description column by the "TBM" designation. All strings are NULL terminated. The description column indicates the source of the information. It is assumed that the user has access to the AVHRR documentation,

"NOAA Polar Orbiter Data Users Guide."

.

Type	Name	Description
CHAR[45]	DataSetName	TBM Data Set Name
ENUM	TotalCopy	TBM Total Copy? EMSC_FALSE or EMSC_TRUE
ENUM	LatitudeSelected	TBM Latitude select option used? EMSC_FALSE or EMSC_TRUE
SHORT	BeginningLatitude	TBM Beginning Latitude
SHORT	EndingLatitude	TBM Ending Latitude
ENUM	LongitudeSelected	TBM Longitude select option used? EMSC_FALSE or EMSC_TRUE
SHORT	BeginningLongitude	TBM Beginning Longitude
SHORT	EndingLongitude	TBM Ending Longitude
ENUM	TimeSelected	TBM Time select option used? EMSC_FALSE or EMSC_TRUE
USHORT	BeginningHour	TBM Beginning Hour of data set
USHORT	BeginningMinute	TBM Beginning Minute of data
USHORT	NumMinutes	TBM Number of minutes in data set
ENUM	AppendedData	TBM Earth location data included?
ENUM[20]	ChannelsSelected	TBM EMSC_FALSE or EMSC_TRUE for 20 channels
CHAR[10]	SpacecraftID	Satellite Name
ENUM	DataType	EIXM_AVHRR_DATATYPE_UNKNOWN, EIXM_AVHRR_LAC, EIXM_AVHRR_GAC, or EIXM_AVHRR_HRPT

ENUM	TipSource	EIXM_AVHRR_TIPSOURCE_UNKNOWN, EIXM_AVHRR_EMBEDDED_TIP, EIXM_AVHRR_STORED_TIP, or EIXM_AVHRR_THIRD_CDA_TIP
TIME	StartTime	Time code from first frame of data processed for the data set
ULONG	NumScans	Number of data scans
TIME	EndTime	Time code from the last frame of data processed
ULONG	ProcessingBlockID	Processing Block ID
UCHAR	CalibrationByte	Ramp/Auto Calibration field
USHORT	NumDataGaps	Number of data gaps
USHORT	NoFrameSyncWordErrorsCount	Number of input data frames with no frame sync word errors
USHORT	TipParityErrorsCount	Number of DACS detected TIP parity errors
USHORT	AuxiliarySyncErrorsCount	Sum of all auxiliary sync errors detected in input data
UCHAR[2]	CalibrationParameterID	Identifies the calibration parameter input data set
ENUM	PseudoNoiseFlag	Normal data = EMSC_FALSE, Pseudo Noise data = EMSC_TRUE
CHAR[10]	DataSource	Receiving station
ENUM	TapeDirection	EIXM_AVHRR_TAPE_DIR_UNKNOWN, EIXM_AVHRR_REVERSE or EIXM_AVHRR_FORWARD
ENUM	DataMode	EIXM_AVHRR_DATAMODE_UNKONWN, EIXM_AVHRR_TESTDATA or EIXM_AVHRR_FLIGHTDATA

DEM Header

The following table defines the contents of the record written to an IMAGINE .img file which has been read from a DEM source. The fields are copied from the Type A and Type C records. All strings are NULL terminated. The description column indicates the source of the information. It is assumed that the user has access to the DEM documentation.

.

Type	Name	Description
CHAR[41]	FileName	DEM quadrangle name
CHAR[41]	FreeText	Free format descriptor field
CHAR	ProcessCode	1=GPM, 2=Manual Profile, 3=DLG2DEM, 4=DCLASS
CHAR[4]	SectionalIndicato	Identifies 1:100000−scale sections
CHAR[5]	MCOriginCode	Mapping center origin code
SHORT	DEMLevelCode	DEM Level
SHORT	ElevationPatternCode	1=regular, 2=random
SHORT	GroundRefSystemCode	0=Geographic, 1=UTM, 2=State Plane
SHORT	Zone	State Plane or UTM Zone
DOUBLE[15]	ProjectionParams	USGS projection parameters
SHORT	PlanimetricUnitCode	0=rad, 1=ft, 2=m, 3=arc−seconds
SHORT	ElevationUnitCode	1=feet, 2=meters
SHORT	CoveragePolygonSides	# polygon sides defining DEM coverage
DOUBLE	SWCornerX	X Coordinate of SW corner
DOUBLE	SWCornerY	Y Coordinate of SW corner
DOUBLE	NWCornerX	X Coordinate of NW corner
DOUBLE	NWCornerY	Y Coordinate of NW corner

DOUBLE	NECornerX	X Coordinate of NE corner
DOUBLE	NECornerY	Y Coordinate of NE corner
DOUBLE	SECornerX	X Coordinate of SE corner
DOUBLE	SECornerY	Y Coordinate of SE corner
DOUBLE	MinElevation	Minimum coverage elevation
DOUBLE	MaxElevation	Maximum coverage elevation
DOUBLE	RotationAngle	CCW angle to local DEM reference system
SHORT	ElevationAccuracyCode	0=unknown, 1=info in Type C record
DOUBLE	SpatialResolutionX	Spatial resolution in X direction
DOUBLE	SpatialResolutionY	Spatial resolution in Y direction
DOUBLE	SpatialResolutionZ	Spatial resolution in Z direction
SHORT	ProfileRows	Number of rows per elevation profile
SHORT	ProfileCols	Number of columns per elevation profile
SHORT	LargestPrimaryContourInterval	Present only if 2 or more primary intervals exist
SHORT	LargestPrimaryIntervalUnits	0=N.A., 1=feet, 2=meters
SHORT	SmallestPrimaryContourInterval	Smallest or only primary contour interval
SHORT	SmallestPrimaryIntervalUnits	1=feet, 2=meters
CHAR[5]	DataSourceDate	YYMM format
CHAR[5]	DataInspRevDate	YYMM format
CHAR	InspectionRevisionFlag	"I" or "R"
SHORT	DataValidationFlag	See USGS manual

SHORT	SuspectVoidAreaFlag	0=none, 1=suspect areas, 2=void areas, 3=suspect and void areas
SHORT	VerticalDatum	1=local mean sea level, 2=NGVD 29, 3=NAVD 88
SHORT	HorizontalDatum	1=NAD 27, 2=WGS 72, 3=WGS 84, 4=NAD 83, 5=Old Hawaii Datum, 6=Puerto Rico Datum, 7=NAD 83 Provisional
SHORT	DataEdition	Primarily DMA Specific field
SHORT	DatumStatsAvailable	1=available, 0=unavailable
SHORT	DatumAccuracyX	In PlanimetricUnitCode units
SHORT	DatumAccuracyY	In PlanimetricUnitCode units
SHORT	DatumAccuracyZ	In ElevationUnitCode units
LONG	DatumSampleSize	0=accuracy assumed to be estimated
SHORT	DemStatsAvailable	1=available, 0=unavailable
SHORT	DemAccuracyX	In PlanimetricUnitCode units
SHORT	DemAccuracyY	In PlanimetricUnitCode units
SHORT	DemAccuracyZ	In ElevationUnitCode units
LONG	DemSampleSize	0=accuracy assumed to be estimated

DOQ Header

The following table defines the contents of the record written to an ERDAS IMAGINE .img file which has been read from a USGS DOQ (Digital Ortho Quadrangle) source. The fields are copied from the four DOQ header records. All strings are NULL terminated. It is assumed that the user has access to the DOQ documentation.

Type	**Name**	**Description**

CHAR[39]	quadrangleName	Authorized quadrangle name
CHAR[3]	quadrant	Quadrangle quadrant (SW, NW, NE, SE)
CHAR[3]	nation1	FIPS nation code, primary nation
CHAR[3]	nation2	FIPS nation code, secondary nation
CHAR[3]	state1	First state in file
CHAR[3]	state2	Second state in file
CHAR[3]	state3	Third state in file
CHAR[3]	state4	Fourth state in file
CHAR[4]	state1county1	First county in first state
CHAR[4]	state1county2	Second county in first state
CHAR[4]	state1county3	Third county in first state
CHAR[4]	state1county4	Fourth county in first state
CHAR[4]	state1county5	Fifth county in first state
CHAR[4]	state2county1	First county in second state
CHAR[4]	state2county2	Second county in second state
CHAR[4]	state2county3	Third county in second state
CHAR[4]	state2county4	Fourth county in second state
CHAR[4]	state2county5	Fifth county in second state
CHAR[4]	state3county1	First county in third state
CHAR[4]	state3county2	Second county in third state
CHAR[4]	state3county3	Third county in third state
CHAR[4]	state3county4	Fourth county in third state
CHAR[4]	state3county5	Fifth county in third state
CHAR[4]	state4county1	First county in fourth state
CHAR[4]	state4county2	Second county in fourth state
CHAR[4]	state4county3	Third county in fourth state

CHAR[4]	state4county4	Fourth county in fourth state
CHAR[4]	state4county5	Fifth county in fourth state
CHAR[24]	descriptiveText	Additional descriptive text
CHAR[5]	producerCode	Mapping center origin code
SHORT	dataOrdering	Direction of lines and samples
LONG	numLines	Number of lines (rows) in image
LONG	numSamples	Number of samples (columns) in image
SHORT	bandTypesAndOrder	Number, types, and order of bands
SHORT	elevationStorage	Elevation storage code
SHORT	bandAndElevationStorage	Method of elevation storage
SHORT	verticalDatum	Vertical datum used
SHORT	primaryHorizDatum	Primary horizontal datum
SHORT	secondaryHorizDatum	Secondary horizontal datum
DOUBLE	rotationAngle	Orthophoto rotation WRT primary datum
SHORT	groundXYRefSystem	Geographic, UTM or State Plane
SHORT	groundXYZone	Zone number if UTM or State Plane
SHORT	groundXYUnits	Map units
DOUBLE[2]	SWPrimaryGroundCoord	Map coords of SW primary quad corner
DOUBLE[2]	NWPrimaryGroundCoord	Map coords of NW primary quad corner
DOUBLE[2]	NEPrimaryGroundCoord	Map coords of NE primary quad corner
DOUBLE[2]	SEPrimaryGroundCoord	Map coords of SE primary quad corner
DOUBLE[6]	primaryCoeff	Primary transform coefficients
DOUBLE[2]	primaryCentroid	X and Y primary centroid

DOUBLE[2]	SWSecondaryGroundCoord	Map coords of SW secondary quad corner
DOUBLE[2]	NWSecondaryGroundCoord	Map coords of NW secondary quad corner
DOUBLE[2]	NESecondaryGroundCoord	Map coords of NE secondary quad corner
DOUBLE[2]	SESecondaryGroundCoord	Map coords of SE secondary quad corner
DOUBLE[6]	secondaryCoeff	Secondary transform coefficients
DOUBLE[2]	secondaryCentroid	X and Y secondary centroid
LONG[2]	SWPrimaryInternalCoord	File coords of SW primary quad corner
LONG[2]	NWPrimaryInternalCoord	File coords of NW primary quad corner
LONG[2]	NEPrimaryInternalCoord	File coords of NE primary quad corner
LONG[2]	SEPrimaryInternalCoord	File coords of SE primary quad corner
LONG[2]	SWSecondaryInternalCoord	File coords of SW secondary quad corner
LONG[2]	NWSecondaryInternalCoord	File coords of NW secondary quad corner
LONG[2]	NESecondaryInternalCoord	File coords of NE secondary quad corner
LONG[2]	SESecondaryInternalCoord	File coords of SE secondary quad corner
DOUBLE[2]	firstPixelPrimary	Primary map coords of (1,1) file pixel
DOUBLE[2]	firstPixelSecondary	Secondary map coords of (1,1) file pixel
SHORT	elevationUnits	DEM elevation units
FLOAT	minElevation	Minimum DEM elevation
FLOAT	maxElevation	Maximum DEM elevation

FLOAT	groundResolutionX	Ground X resolution of DEM
FLOAT	groundResolutionY	Ground Y resolution of DEM
FLOAT	groundResolutionZ	Ground Z resolution of DEM
FLOAT	pixelResolutionX	X resolution of orthophoto
FLOAT	pixelResolutionY	Y resolution of orthophoto
FLOAT	pixelResolutionZ	Z resolution of orthophoto
LONG	largestPrimaryContourInterval	Largest interval if DEM derived from graphic
SHORT	largestPrimaryContourIntervalUnits	Units of Largest interval
LONG	smallestPrimaryContourInterval	Smallest interval if DEM derived from graphic
SHORT	smallestPrimaryContourIntervalUnits	Units of Smallest interval
SHORT	suspectAndVoidAreas	Suspect and void areas in elevation or orthophoto data
FLOAT	horizDoqAccuracy	RMSE of image control points for X−Y
FLOAT	verticalDoqAccuracy	RMSE of primary DEM used
SHORT	numDoqHorizTestPoints	Number of DOQ horizontal test points
SHORT	pixelProcessingAlgorithm	Resampling method
CHAR[25]	productionSystem	Description of hardware & software used
TIME	productionDate	Production date
CHAR[25]	filmType	Manufacturer and ID of film type
CHAR[25]	sourcePhotoID	Descrip. of photo, agency, roll #, etc.
SHORT	mosaickedImage	Number of chips composing image
ENUM	leafOff	Leaves off trees FALSE or TRUE
TIME	sourcePhotoDate	Date of source photo

FLOAT	focalLength	Calibrated camera focal length in mm
LONG	sourcePhotoFlyingHeight	Nominal flying height of photograph
CHAR[25]	scannerType	Scanner description
FLOAT[2]	scanningResolution	(x,y) Aperture resolution, microns
FLOAT[2]	scannerSamplingResolution	(x,y) Sampling resolution, microns
SHORT	radiometricResolution	8 or 16 bits
FLOAT	resampledResolution	Resampled resolution

DTED Header

The following table defines the contents of the record written to an IMAGINE .img file which has been read from a DTED source. The fields are copied from the File Header (HDR), User Header (UHL), Data Set Identification (DSI), and Accuracy Description (ACC) records. The description is preceded by the record indicator from which the field was derived. All strings are NULL terminated. The description column indicates the source of the information. It is assumed that the user has access to the DTED documentation.

.

Type	**Name**	**Description**
CHAR[18]	FileName	HDR File name
TIME	CreationDate	HDR Creation date of tape
TIME	ExpirationDate	HDR Expiration date of tape
CHAR[4]	SecurityCode	UHL Security Code
CHAR[13]	UHL_ReferenceNumber	UHL Unique reference number
SHORT	MultipleAccuracy	UHL 0=Single, 1=Multiple
CHAR	SecurityClassification	DSI Security classification code
CHAR[28]	SecurityHandlingDescription	DSI Security handling description
CHAR[6]	SeriesDesignator	DSI DMA series designator
CHAR[16]	DSI_ReferenceNumber	DSI Unique reference number

SHORT	DataEditionNumber	DSI Data edition number
CHAR	MatchMergeVersion	DSI Match/Merge version
CHAR[5]	MaintenanceDate	DSI Maintenance date (YYMM)
CHAR[5]	MatchMergeDate	DSI Match/Merge date (YYMM)
CHAR[9]	ProducerCode	DSI Producer code
CHAR[10]	ProductSpecStockNum	DSI Product specification stock number
SHORT	ProductSpecAmendmentNum	DSI Product spec. amendment number
CHAR[5]	ProductSpecDate	DSI Product specification date (YYMM)
CHAR[4]	VerticalDatum	DSI Vertical datum
CHAR[6]	HorizontalDatumCode	DSI Horizontal datum code
CHAR[11]	DigitizingCollectionSystem	DSI Digitizing collection system
CHAR[5]	CompilationDate	DSI Compilation date (YYMM)
DOUBLE	OriginLatitude	DSI Latitude of data origin
DOUBLE	OriginLongitude	DSI Longitude of data origin
DOUBLE	SWCornerLatitude	DSI Latitude of SW corner of data
DOUBLE	SWCornerLongitude	DSI Longitude of SW corner of data
DOUBLE	NWCornerLatitude	DSI Latitude of NW corner of data
DOUBLE	NWCornerLongitude	DSI Longitude of NW corner of data
DOUBLE	NECornerLatitude	DSI Latitude of NE corner of data
DOUBLE	NECornerLongitude	DSI Longitude of NE corner of data
DOUBLE	SECornerLatitude	DSI Latitude of SE corner of data
DOUBLE	SECornerLongitude	DSI Longitude of SE corner of data
DOUBLE	OrientationAngle	DSI Clockwise orientation angle of data with respect to true north
DOUBLE	LatitudeInterval	DSI Latitude interval in tenths of seconds between rows of elevation values
DOUBLE	LongitudeInterval	DSI Longitude interval in tenths of seconds between rows of elevation values

SHORT	NumLatitudeLines	DSI Number of latitude lines
SHORT	NumLongitudeLines	DSI Number of longitude lines
SHORT	PartialCellIndicator	DSI Partial Cell Indicator
SHORT	AbsoluteHorizontalAccuracy	ACC Absolute horizontal accuracy of product
SHORT	AbsoluteVerticalAccuracy	ACC Absolute vertical accuracy of product
SHORT	PointToPointHorizAcc	ACC Point−to−Point horizontal accuracy of product
SHORT	PointToPointVertAcc	ACC Point−to−Point vertical accuracy of product

MSS Header

The following table defines the contents of the record written to an IMAGINE .img file which has been read from a MSS source. The fields are copied from the Tape Directory (TD), Header (HDR), and Annotation (ANT) records. The description is preceded by the record indicator from which the field was derived. All strings are NULL terminated. The description column indicates the source of the information. It is assumed that the user has access to the MSS documentation.

Type	**Name**	**Description**
CHAR	MissionNum	TD Byte 8
TIME	TapeCreationDate	TD Bytes 12−16
USHORT	NumVolumes	TD Byte 20
ENUM	format	TD Byte 31 NONE, BIL, BIP or BSQ
USHORT	RecordLength	TD Bytes 32−33
ENUM	SourceHDT	TD Byte 34 FALSE or TRUE

CHAR[19]	SceneID	TD Bytes 35−52
CHAR[8]	WRSDesignator	HDR Bytes 20−26
CHAR[4]	SensorID	HDR Bytes 37−39
USHORT	OrbitNum	HDR Bytes 47−48
USHORT[26]	ActiveDetectorStatus	HDR Bytes 49−52 1 = Detector is Active
USHORT	ActiveDetectorCount	HDR Byte 57
USHORT	UncorrectedPixelsPerLine	HDR Byte 58
USHORT	WRSScanLineNum	HDR Bytes 73−74
USHORT	WRSPixelNum	HDR Bytes 75−76
USHORT	NumAnnotationRecords	HDR Bytes 101−102
USHORT	NumAncillaryRecords	HDR Bytes 105−106
ENUM	GeoCorrectionsApplied	HDR Byte 107 FALSE or TRUE
ENUM	GeoCorrectionDataPresent	HDR Byte 108 FALSE or TRUE
ENUM	RadiometricCorrectionsApplied	HDR Byte 109 FALSE or TRUE
ENUM	RadiometricCorrectionDataPresent	HDR Byte 110 FALSE or TRUE
ENUM	ImageDataFormat	HDR Byte 117 UNKNOWN, UNFRAMED_RECTANGULAR_IMAGE, FRAMED_RECTANGULAR_IMAGE, or FRAMED_SQUARE_IMAGE
USHORT	LineInterleavingCount	HDR Byte 121
USHORT	BitsPerPixel	HDR Byte 122

ENUM	ResamplingApplied	HDR Byte 123 NONE, CUBIC_CONVOLUTION or NEAREST_NEIGHBOR
SHORT	WRSImageCenterOffset	HDR Byte 125−126
USHORT	PixelsPerScanLine	HDR Byte 131−132
USHORT	NumUsableImages	HDR Byte 135
USHORT	NumTrailerRecords	HDR Byte 144−145
ENUM	DayPass	HDR Byte 151 FALSE (Night pass) or TRUE (Day pass)
ENUM	CalWedgeMode	HDR Byte 162 UNKNOWN, LOW_GAIN_LINEAR_TRANSMISSION, LOW_GAIN_COMPRESSED_TRANSMISSION, HIGH_GAIN_LINEAR_TRANSMISSION, HIGH_GAIN_COMPRESSED_TRANSMISSION
USHORT	TempRegPt1CurImgScanLine	HDR Bytes 183−184
USHORT	TempRegPt1CurImgPixelNum	HDR Bytes 185−186
USHORT	TempRegPt1RefImgScanLine	HDR Bytes 187−188
USHORT	TempRegPt1RefImgPixelNum	HDR Bytes 189−190
USHORT	TempRegPt2CurImgScanLine	HDR Bytes 191−192
USHORT	TempRegPt2CurImgPixelNum	HDR Bytes 193−194
USHORT	TempRegPt2RefImgScanLine	HDR Bytes 195−196
USHORT	TempRegPt2RefImgPixelNum	HDR Bytes 197−198
USHORT	TempRegPt3CurImgScanLine	HDR Bytes 199−200
USHORT	TempRegPt3CurImgPixelNum	HDR Bytes 201−202
USHORT	TempRegPt3RefImgScanLine	HDR Bytes 203−204
USHORT	TempRegPt3RefImgPixelNum	HDR Bytes 205−206

USHORT	TempRegPt4CurImgScanLine	HDR Bytes 207−208
USHORT	TempRegPt4CurImgPixelNum	HDR Bytes 209−210
USHORT	TempRegPt4RefImgScanLine	HDR Bytes 211−212
USHORT	TempRegPt4RefImgPixelNum	HDR Bytes 213−214
USHORT	OverlapMark1ScanLine	HDR Bytes 215−216
USHORT	OverlapMark1PixelNum	HDR Bytes 217−218
USHORT	OverlapMark2ScanLine	HDR Bytes 219−220
USHORT	OverlapMark2PixelNum	HDR Bytes 221−222
USHORT	OverlapMark3ScanLine	HDR Bytes 223−224
USHORT	OverlapMark3PixelNum	HDR Bytes 225−226
USHORT	OverlapMark4ScanLine	HDR Bytes 227−228
USHORT	OverlapMark4PixelNum	HDR Bytes 229−230
USHORT	OverlapMarkPixelOffset	HDR Byte 231
USHORT	GeoModelQualityAssessment	HDR Byte 232
USHORT	NumTickMarksTop	HDR Byte 233
USHORT	NumTickMarksLeft	HDR Byte 234
USHORT	NumTickMarksRight	HDR Byte 235
USHORT	NumTickMarksBottom	HDR Byte 236
ENUM	EdipsContrastEnhancement	HDR Byte 3583 FALSE or TRUE
ENUM	EdipsAtmScatterCompensation	HDR Byte 3584 FALSE or TRUE
ENUM	EdipsEdgeEnhancement	HDR Byte 3585 FALSE or TRUE
USHORT[8]	DataPresentByBand	HDR Byte 3586 1 = Data Present
FLOAT	FormatCenterLatitude	ANT Bytes 17−22
FLOAT	FormatCenterLongitude	ANT Bytes 24−31

CHAR[10]	NominalPathRowID	ANT Bytes 32−40
FLOAT	NominalLatitude	ANT Bytes 43−48
FLOAT	NominalLongitude	ANT Bytes 50−57
CHAR[8]	SpectralBandIDCode	ANT Bytes 58−64
ENUM	TransmissionType	ANT Byte 66 UNKNOWN, DIRECT or STORED_PLAYBACK
CHAR[15]	SunAngles	ANT Bytes 68−81
ENUM	CorrectionType	ANT Byte 82 NONE, SYSTEM_CORRECTION, GEOMETRIC_GCP_CORRECTION, or RELATIVE_GCP_CORRECTION
CHAR[16]	ImageScale	ANT Byte 83 "185 km x 185 km", " 99 km x 99 km", or "185 km x 170 km"
Eixm_Mss− MapProjection	MapProjection	ANT Byte 84 NONE, UTM, POLAR_STEREOGRAPHIC, HOTINE_OBLIQUE_MERCATOR, SPACE_OBLIQUE_MERCATOR, or LAMBERT
ENUM	EphemerisType	ANT Byte 87 UNKNOWN, PREDICTIVE, or DEFINITIVE
ENUM	ProcessingProcedure	ANT Byte 89 UNKNOWN, ABNORMAL, or NORMAL

ENUM	SensorGain	ANT Byte 91 UNKNOWN, HIGH_GAIN or LOW_GAIN
CHAR[14]	AgencyProject	ANT Bytes 94−106
CHAR[16]	FrameID	ANT Bytes 107−121
struct TickMark[16]	TopTickMark	ANT Bytes 405−548 See TickMark Structure table below.
struct TickMark[25]	LeftTickMark	ANT Bytes 803−964 See TickMark Structure table below.
struct TickMark[25]	RightTickMark	ANT Bytes 1201−1263 See TickMark Structure table below.
struct TickMark[16]	BottomTickMark	ANT Bytes 1599−1760 See TickMark Structure table below.

Type	Name	Description
USHORT	position	Number of pixels from top or left side
CHAR[8]	label	Tick Mark Label

TickMark Structure

SPOT Header

The following table defines the contents of the record written to an IMAGINE .img file which has been read from a SPOT source. The fields are copied from the SPOT Level 1A, 1B, and 2 headers, SPOTVIEW/GEOSPOT (GIS) headers, and the Canadian SPOT (RADARSAT) headers. If the value is a string, it is left unchanged. If the value is a number, it is converted from ASCII to binary. The description column indicates the source of the information. It is assumed that the user has access to the various SPOT documentation.

MAX_INT: Maximum integer. It is the default uninitialized value for certain fields.

MAX_DBL: Maximum double. It is the default uninitialized value for certain fields.

SP: SPOT Level 1A, 1B, and 2.

SV: SPOTVIEW 1.5 (GIS format).

GS: GEOSPOT 4.0 (GIS format).[1]

CS: Canadian SPOT.[2]

1. GIS−GEOSPOT Format Reference Manual

2. Canadian SPOT documentation (RADARSAT).

⊕ The Canadian SPOT header fields are not completely defined in this document. This will be completed in a future edition of On−Line Help.

struct SpotHeader

Type	Name	Description
CHAR[15]	scnname	SP: based on Fields 13 and 14 of file 2 record 1. SV: This field is empty. GS: This field is empty. CS:
DOUBLE	scncenlat	SP: Field 13 of file 2 record 2. SV: This field is 0.0. GS: This field is 0.0. CS:
DOUBLE	scncenlon	SP: Field 14 of file 2 record 2. SV: This field is 0.0. GS: This field is 0.0. CS:
DOUBLE	scncor1lat	SP: Field 17 of file 2 record 2. SV: This field is 0.0. GS: tag NW_LAT in REP header file. CS:
DOUBLE	scncor1lon	SP: Field 18 of file 2 record 2. SV: This field is 0.0. GS: tag NW_LON in REP header file. CS:

DOUBLE	scncor2lat	SP: Field 21 of file 2 record 2. SV: This field is 0.0. GS: tag SW_LAT in REP header file. CS:
DOUBLE	scncor2lon	SP: Field 22 of file 2 record 2. SV: This field is 0.0. GS: tag SW_LON in REP header file. CS:
DOUBLE	scncor3lat	SP: Field 25 of file 2 record 2. SV: This field is 0.0. GS: tag NE_LAT in REP header file. CS:
DOUBLE	scncor3lon	SP: Field 26 of file 2 record 2. SV: This field is 0.0. GS: tag NE_LON in REP header file. CS:
DOUBLE	scncor4lat	SP: Field 29 of file 2 record 2. SV: This field is 0.0. GS: tag SE_LAT in REP header file. CS:
DOUBLE	scncor4lon	SP: Field 30 of file 2 record 2. SV: This field is 0.0. GS: tag SE_LON in REP header file. CS:
LONG	scncenline	SP: Field 15 of file 2 record 2. SV: This field is 0. GS: This field is 0.0. CS:
LONG	scncenpixel	SP: Field 16 of file 2 record 2. SV: This field is 0. GS: This field is 0.0. CS:

LONG	scncor1line	SP: Field 19 of file 2 record 2. SV: This field is 0. GS: tag NW_Y_PIXEL in HDR header file. CS:
LONG	scncor1pixel	SP: Field 20 of file 2 record 2. SV: This field is 0. GS: tag NW_X_PIXEL in HDR header file. CS:
LONG	scncor2line	SP: Field 23 of file 2 record 2. SV: This field is 0. GS: tag SW_Y_PIXEL in HDR header file. CS:
LONG	scncor2pixel	SP: Field 24 of file 2 record 2. SV: This field is 0. GS: tag SW_X_PIXEL in HDR header file. CS:
LONG	scncor3line	SP: Field 27 of file 2 record 2. SV: This field is 0. GS: tag NE_Y_PIXEL in HDR header file. CS:
LONG	scncor3pixel	SP: Field 28 of file 2 record 2. SV: This field is 0. GS: tag NE_X_PIXEL in HDR header file. CS:
LONG	scncor4line	SP: Field 31 of file 2 record 2. SV: This field is 0. GS: tag SE_Y_PIXEL in HDR header file. CS:
LONG	scncor4pixel	SP: Field 32 of file 2 record 2. SV: This field is 0. GS: tag NE_X_PIXEL in HDR header file. CS:

DOUBLE	scnorientangle	SP: Field 35 of file 2 record 2. SV: This field is 0.0. GS: This field is 0.0. CS:
DOUBLE	angleofinc	SP: Field 36 of file 2 record 2.[a] SV: This field is 0.0. GS: This field is 0.0. CS:
DOUBLE	sunazimuth	SP: Field 37 of file 2 record 2. SV: This field is 0.0. GS: This field is 0.0. CS:
DOUBLE	sunelevation	SP: Field 38 of file 2 record 2. SV: This field is 0.0. GS: This field is 0.0. CS:
CHAR[17]	scncentime	SP: Field 40 of file 2 record 2. SV: This field is empty. GS: This field is empty. CS:
CHAR[4]	sensorid	SP: Field 42 of file 2 record 2. SV: This field is empty. GS: This field is empty. CS:
CHAR[3]	specmode	SP: Field 43 of file 2 record 2. SV: This field is empty. GS: This field is empty. CS:
LONG	revolutionnum	SP: Field 44 of file 2 record 2. SV: This field is 0. GS: This field is 0. CS:

CHAR[2]	playback	SP: Field 47 of file 2 record 2. SV: This field is empty. GS: This field is empty. CS:
CHAR[9]	preprocesslevel	SP: Field 55 of file 2 record 2. SV: tag PRE_PROCESS_LEVEL in HDR header file or tag CORRECTION_LEVEL in HDR header file. GS: tag PRE_PROCESS_LEVEL in HDR header file or tag CORRECTION_LEVEL in HDR header file. CS:
ENUM	correction	SP: based on Field 56 of file 2 record 2. SV: This field is EMSC_FALSE. GS: This field is EMSC_FALSE. CS:
LONG	deconvolution	SP: Field 57 of file 2 record 2. SV: This field is 0. GS: This field is 0. CS:
CHAR	resampling	SP: Field 58 of file 2 record 2. SV: This field is empty. GS: This field is empty. CS:
LONG	ulxmap	SP: This field is 0. SV: tag ULXMAP from header file. GS: tag NW_X_COORD in REP header file. CS:
LONG	ulymap	SP: This field is 0. SV: tag ULYMAP from header file. GS: tag NW_Y_COORD in REP header file. CS:

LONG	lrxmap	SP: This field is 0. SV: tag LRXMAP from header file. GS: tag SE_X_COORD in REP header file. CS:
LONG	lrymap	SP: This field is 0. SV: tag LRYMAP from header file. GS: tag SE_Y_COORD in REP header file. CS:
LONG	xpixelsize	SP: Field 59 of file 2 record 2. SV: tag XDIM in HDR header file. GS: tag XDIM in HDR header file. CS:
LONG	ypixelsize	SP: Field 60 of file 2 record 2. SV: tag YDIM in HDR header file. GS: tag YDIM in HDR header file. CS:
DOUBLE	mapxpixelsize	SP: Field 63 of file 2 record 2. SV: tag XDIM from header file. GS: tag XDIM in HDR header file. CS: This field is 0.0.
DOUBLE	mapypixelsize	SP: Field 62 of file 2 record 2. SV: tag YDIM from header file. GS: tag YDIM in HDR header file. CS: This field is 0.0.
CHAR[32]	mapprojid	SP: Field 61 of file 2 record 2. SV: This field is empty. GS: This field is empty. CS:
LONG	numcols	SP: Field 50 of file 2 record 2. SV: tag NCOLS from header file. GS: tag NCOLS in HDR header file. CS:

LONG	numlines	SP: Field 51 of file 2 record 2. SV: tag NROWS from header file. GS: tag NROWS in HDR header file. CS:
LONG	numbands	SP: Field 9 of file 3 record 1. SV: tag NBANDS from header file. GS: tag NBANDS in HDR header file. CS:
CHAR[4]	interleave	SP: Field 52 of file 2 record 2. SV: This field is empty. GS: tag LAYOUT in HDR header file. CS:
CHAR[33]	cartcoord	SP: Field 72 of file 2 record 2. SV: This field is empty. GS: This field is empty. CS:
CHAR[16]	suncal	SP: Field 79 of file 2 record 2. SV: This field is empty. GS: This field is empty. CS:
CHAR[8]	mapunits	SP: This field is empty. SV: tag MAPUNITS from header file. GS: tag MAPUNITS in HDR header file. CS:
CHAR[35]	mapproj	SP: This field is empty. SV: tag PROJECTION from header file. GS: tag PROJ_ID in REP header file. CS:
LONG	utmzone	SP: This field is empty. SV: tag UTM_ZONE from header file. GS: tag PROJ_ZONE in REP header file. CS:

LONG	utmeast	SP: This field is empty. SV: tag UL_UTMEAST from header file. GS: This field is MAX_INT. CS:
LONG	utmnorth	SP: This field is empty. SV: tag UL_UTMNORTH from header file. GS: This field is MAX_INT. CS:
LONG	stplaneeast	SP: This field is empty. SV: tag UL_STPLANE_EAST from header file. GS: This field is MAX_INT. CS:
LONG	stplanenorth	SP: This field is empty. SV: tag UL_STPLANE_NORTH from header file. GS: This field is MAX_INT. CS:
LONG	stplanecode	SP: This field is empty. SV: tag ST_PROJ_CODE from header file. GS: tag PROJ_ZONE in REP header file. CS:
CHAR[6]	hdatum	SP: This field is empty. SV: tag HORIZONTAL_DATUM or tag DATUM from header file. GS: tag HORIZ_DATUM in HDR header file. CS: This field is empty.
CHAR[4]	rgborder	SP: Defaults to "123". SV: tag BAND_RGB from header file. GS: tag BAND_RBG in HDR header file. CS: Defaults to "123".
LONG	GRSk	SP: Field 9 of file 2 record 2. SV: This field is 0. GS: This field is 0. CS:

LONG	GRSj	SP: Field 9 of file 2 record 2. SV: This field is 0. GS: This field is 0. CS:
DOUBLE	satlongnadir	SP: Field 34 of file 2 record 2. SV: This field is 0.0. GS: This field is 0.0. CS:
DOUBLE	satlatnadir	SP: Field 33 of file 2 record 2. SV: This field is 0.0. GS: This field is 0.0. CS:
CHAR[17]	missionid	SP: Field 68 of file 2 record 2. SV: This field is empty. GS: This field is empty. CS:
DOUBLE	pointingmirror	SP: Field 45 of file 2 record 2. SV: This field is 0.0. GS: This field is 0.0. CS:
CHAR[7]	telmode	SP: Field 46 of file 2 record 2. SV: This field is empty. GS: This field is empty. CS:
CHAR[6]	satelliteid	SP: Field 41 of file 2 record 2. SV: This field is empty. GS: This field is empty. CS:
DOUBLE[9]	xcoord	SP: Field 9 of file 2 record 3[b]. SV: These fields contain 0.0s. GS: These fields contain 0.0s. CS:

DOUBLE[9]	ycoord	SP: Field 9 of file 2 record 3[b]. SV: These fields contain 0.0s. GS: These fields contain 0.0s. CS:
DOUBLE[9]	zcoord	SP: Field 9 of file 2 record 3[b]. SV: These fields contain 0.0s. GS: These fields contain 0.0s. CS:
DOUBLE[9]	xvelocityvector	SP: Field 9 of file 2 record 3[b]. SV: These fields contain 0.0s. GS: These fields contain 0.0s. CS:
DOUBLE[9]	yvelocityvector	SP: Field 9 of file 2 record 3[b]. SV: These fields contain 0.0s. GS: These fields contain 0.0s. CS:
DOUBLE[9]	zvelocityvector	SP: Field 9 of file 2 record 3[b]. SV: These fields contain 0.0s. GS: These fields contain 0.0s. CS:
LONG[9]	julianday	SP: Field 9 of file 2 record 3[b]. SV: These fields contain 0s. GS: These fields contain 0s. CS:
LONG[9]	timeofday	SP: Field 9 of file 2 record 3[b]. SV: This field contains 0s. GS: CS:
CHAR[27]	scenecentertime	SP: Field 12 of file 2 record 3. SV: This field is empty. GS: This field is empty. CS:

LONG[73]	rawimageline	SP: Field 14 of file 2 record 3[b]. SV: These fields contain 0s. GS: These fields contain 0s. CS:
LONG[73]	yawspeed	SP: Field 14 of file 2 record 3[b]. SV: These fields contain 0s. GS: These fields contain 0s. CS:
LONG[73]	avgrollspeed	SP: Field 14 of file 2 record 3[b]. SV: These fields contain 0s. GS: These fields contain 0s. CS:
LONG[73]	avgpitchspeed	SP: Field 14 of file 2 record 3[b]. SV: This field contains 0s. GS: CS:
DOUBLE[4]	lookangles	SP: Field 15−18 of file 2 record 3[b]. SV: This field contains 0.0s. GS: CS:
DOUBLE[4]	polynomial1	SP: Field 20 of file 2 record 3[b]. SV: These fields contain 0.0s. GS: These fields contain 0.0s. CS:
DOUBLE[4]	polynomial2	SP: Field 21 of file 2 record 3[b]. SV: These fields contain 0.0s. GS: These fields contain 0.0s. CS:
DOUBLE[4]	polynomial3	SP: Field 22 of file 2 record 3[b]. SV: These fields contain 0.0s. GS: These fields contain 0.0s. CS:

DOUBLE[4]	polynomial4	SP: Field 23 of file 2 record 3[b]. SV: These fields contain 0.0s. GS: These fields contain 0.0s. CS:
DOUBLE[4]	polynomial5	SP: Field 24 of file 2 record 3[b]. SV: These fields contain 0.0s. GS: These fields contain 0.0s. CS:
CHAR[17]	level2quality	SP: Field 77 of file 2 record 2. SV: This field is empty. GS: This field is emtpy. CS:
CHAR[17]	dateofdarkcal	SP: Field 78 of file 2 record 2. SV: This field is empty. GS: This field is empty. CS:
LONG	ephemerislength	SP: Field 84 of file 2 record 2. SV: This field is 0. GS: This field is 0. CS:
LONG	ephemerisnumber	SP: Field 83 of file 2 record 2. SV: This field is 0. GS: This field is 0. CS:
LONG	numberofgcps	SP: Field 64 of file 2 record 2. SV: This field is 0. GS: This field is 0. CS:
CHAR[17]	GRSdesignator	SP: Field 66 of file 2 record 2. SV: This field is empty. GS: This field is empty. CS:

CHAR[17]	referencesensorid	SP: Field 69 of file 2 record 2. SV: This field is empty. GS: This field is empty. CS: This field is empty.
CHAR[17]	referencespecmode	SP: Field 70 of file 2 record 2. SV: This field is empty. GS: This field is empty. CS: This field is empty.
CHAR[17]	referenceprocesslevel	SP: Field 71 of file 2 record 2. SV: This field is empty. GS: This field is empty. CS: See field "**preprocesslevel**".
LONG	numberlostlines	SP: Field 74 of file 2 record 2. SV: This field is 0.0. GS: This field is 0.0. CS:
LONG	numberlostdetectors	SP: Field 75 of file 2 record 2. SV: This field is 0. GS: This field is 0. CS:
CHAR[17]	sceneidentification	SP: Field 10 of file 2 record 2. SV: This field is empty. GS: This field is empty. CS: This field is empty.
DOUBLE	GRSlatdelta	SP: Field 11 of file 2 record 2. SV: This field is 0.0. GS: This field is 0.0. CS:
DOUBLE	GRSlondelta	SP: Field 12 of file 2 record 2. SV: This field is 0.0. GS: This field is 0.0. CS:

LONG	gainnumber	SP: Field 48 of file 2 record 2. SV: This field is 0. GS: This field is 0.0. CS:
DOUBLE	xpixelsized	See field "**xpixelsize**".
DOUBLE	ypixelsized	See field "**ypixelsize**".
DOUBLE	urxmap	SP: This field is 0.0. SV: This field is 0.0. GS: tag NE_X_COORD in REP header file. CS:
DOUBLE	urymap	SP: This field is 0.0. SV: This field is 0.0. GS: tag NE_Y_COORD in REP header file. CS:
DOUBLE	llxmap	SP: This field is 0.0. SV: This field is 0.0. GS: tag SW_X_COORD in REP header file. CS:
DOUBLE	llymap	SP: This field is 0.0. SV: This field is 0.0. GS: tag SW_Y_COORD in REP header file. CS:
CHAR *	datum[]	See field "**hdatum**".
struct	GeoSpotHeader	See GeoSpot Header Table below.
struct	SpotCanHeader	See **Spot Canadian Header Table** below.

a. First character of string is either ‘L’ or ‘R‘ depending on how satellite passed over scanned area (L = negative; R = positive)

b. Refer to the SPOT Standard CCT Format Document

GeoSpot Header Table[a]

Type	Name	Description

ENUM	geospotformat	SP: This field is EMSC_FALSE. SV: This field is EMSC_TRUE. GS: This field is EMSC_TRUE. CS: This field is EMSC_FALSE.
CHAR *	geospotversion[]	SP: This field is empty. SV: This field is "1.5". GS: This field is empty.[b] CS: This field is empty.
LONG	bandrowbytes	SP: This field is 0.0. SV: tag BANDROWBYTES in header file. GS: tag BANDROWBYTES in HDR header file. CS: This field is 0.0.
CHAR	byteorder	SP: This field is empty. SV: tag BYTEORDER in header file. GS: tag BYTEORDER in HDR header file. CS: This field is empty.
LONG	numbits	SP: This field is 0. SV: tag NBITS in header file. GS: tag NBITS in HDR header file. CS: This field is 0.
CHAR *	correction_level[]	See field "**preprocesslevel**".
DOUBLE	deltax_origin	SP: This field is 0.0. SV: This field is −MAX_DBL. GS: CS: This field is 0.0.
DOUBLE	deltay_origin	SP: This field is 0.0. SV: This field is −MAX_DBL. GS: CS: This field is 0.0.
CHAR *	contact_info[]	SP: This field is empty. SV: This field is empty. GS: tag CONTACT_INFO in REP header file. CS: This field is empty.

CHAR *	easting_size_units	SP: This field is empty. SV: This field is empty. GS: tag EASTING_SIZE argument 2 in REP header file. CS: This field is empty.
CHAR *	northing_size_units	SP: This field is empty. SV: This field is empty. GS: tag NORTHING_SIZE argument 2 in REP header file. CS: This field is empty.
DOUBLE	grid_declination	SP: This field is 0.0. SV: This field is 0.0. GS: tag GRID_DECLINATION in REP header file. CS: This field is 0.0.
CHAR *	job_id[]	SP: This field is empty. SV: This field is empty. GS: tag JOB_ID in REP header file. CS: This field is empty.
CHAR *	location[]	SP: This field is empty. SV: This field is empty. GS: tag LOCATION in REP header file. CS: This field is empty.
DOUBLE	magnetic_declination	SP: This field is 0.0. SV: This field is 0.0. GS: tag MAGNETIC_DECLINATION in REP header file. CS: This field is 0.0.
DOUBLE	mag_decl_annual_chg	SP: This field is 0.0. SV: This field is 0.0. GS: tag MAG_DECL_ANNUAL_CHG in REP header file. CS: This field is 0.0.

CHAR *	mag_decl_date[]	SP: This field is empty. SV: This field is empty. GS: tag MAG_DECL_DATE in REP header file. CS: This field is empty.
CHAR *	map_name[]	SP: This field is empty. SV: This field is empty. GS: tag MAP_NAME in REP header file. CS: This field is empty.
CHAR *	map_number[]	SP: This field is empty. SV: This field is empty. GS: CS: This field is empty.
CHAR *	meridian_name[]	SP: This field is empty. SV: This field is empty. GS: tag MERIDIAN_NAME in REP header file. CS: This field is empty.
DOUBLE	meridian_origin	SP: This field is 0.0. SV: This field is 0.0. GS: tag MERIDIAN_ORIGIN in REP header file. CS: This field is 0.0.
DOUBLE	origin_x_coord	SP: This field is 0.0. SV: This field is 0.0. GS: tag ORIGIN_X_COORD in REP header file. CS: This field is 0.0.
DOUBLE	origin_y_coord	SP: This field is 0.0. SV: This field is 0.0. GS: tag ORIGIN_Y_COORD in REP header file. CS: This field is 0.0.

CHAR *	production_date[]	SP: This field empty. SV: This field is empty. GS: tag PRODUCTION_DATE in REP header file. CS: This field is empty.
CHAR *	product_id[]	SP: This field empty. SV: This field is empty. GS: tag PRODUCT_ID in REP header file. CS: This field is empty.
CHAR *	proj_code[]	SP: This field empty. SV: This field is empty. GS: tag PROJ_CODE in REP header file. CS: This field is empty.
CHAR *	proj_id[]	SP: This field empty. SV: This field is empty. GS: tag PROJ_ID in REP header file. CS: This field is empty.
DOUBLE	proj_lat_true_scale	SP: This field is 0.0. SV: This field is 0.0. GS: tag PROJ_LAT_TRUE_SCALE in REP header file. CS: This field is 0.0.
DOUBLE	proj_meridian	SP: This field is 0.0. SV: This field is −MAX_DBL. GS: tag PROJ_MERIDIAN in REP header file. CS: This field is 0.0.
DOUBLE	proj_parallel	SP: This field is 0.0. SV: This field is −MAX_DBL. GS: tag PROJ_PARALLEL in REP header file. CS: This field is 0.0.

DOUBLE	proj_scale_factor	SP: This field is 0.0. SV: This field is 0.0. GS: tag PROJ_SCALE_FACTOR in REP header file. CS: This field is 0.0.
DOUBLE	proj_std_parallel_I	SP: This field is 0.0. SV: This field is −MAX_DBL GS: tag PROJ_STD_PARALLEL_I in REP header file. CS: This field is 0.0.
DOUBLE	proj_std_parallel_II	SP: This field is 0.0. SV: This field is −MAX_DBL GS: tag PROJ_STD_PARALLEL_II in REP header file. CS: This field is 0.0.
LONG	proj_zone	SP: This field is 0. SV: This field is 0.0. GS: tag PROJ_ZONE in REP header file. CS: This field is 0.
DOUBLE	raster_x_origin	SP: This field is 0.0. SV: This field is 0.0. GS: tag RASTER_X_ORIGIN in REP header file. CS: This field is 0.0.
DOUBLE	raster_x_size	SP: This field is 0.0. SV: This field is 0.0. GS: tag RASTER_X_SIZE in REP header file. CS: This field is 0.0.
DOUBLE	raster_y_origin	SP: This field is 0.0. SV: This field is 0.0. GS: tag RASTER_Y_ORIGIN in REP header file. CS: This field is 0.0.

DOUBLE	raster_y_size	SP: This field is 0.0. SV: This field is 0.0. GS: tag RASTER_Y_SIZE in REP header file. CS: This field is 0.0.
DOUBLE	sheet_rect_elevation	SP: This field is 0.0. SV: This field is empty. GS: tag SHEET_RECT_ELEVATION in REP header file. CS: This field is 0.0.
CHAR *	sheet_rect_type[]	SP: This field empty. SV: This field is empty. GS: tag SHEET_RECT_TYPE in REP header file. CS: This field is empty.
DOUBLE	spheroid_flat	SP: This field is 0.0. SV: This field is 0.0. GS: tag SPHEROID_FLAT in REP header file. CS: This field is 0.0.
DOUBLE	spheroid_maj_axis	SP: This field is 0.0. SV: This field is 0.0. GS: tag SPHEROID_MAJ_AXIS in REP header file. CS: This field is 0.0.
DOUBLE	spheroid_min_axis	SP: This field is 0.0. SV: This field is 0.0. GS: tag SPHEROID_MIN_AXIS in REP header file. CS: This field is 0.0.
CHAR *	spheroid_name[]	SP: This field empty. SV: This field is empty. GS: tag SPHEROID_NAME in REP header file. CS: This field is empty.

a. Compiled from SPOTVIEW 1.5 and GEOSPOT 4.0 documentation.

b. If geospotversion is EMSC_TRUE, an empty field implies GEOSPOT version 4.0.

Spot Canadian Header Table

Type	Name	Description
DOUBLE	nom_interpixel_dist	SP: This field is 0.0. SV: This field is 0.0. GS: This field is 0.0. CS:
DOUBLE	mon_interline_dist	SP: This field is 0.0. SV: This field is 0.0. GS: This field is 0.0. CS:
DOUBLE	image_skew	SP: This field is 0.0. SV: This field is 0.0. GS: This field is 0.0. CS:
DOUBLE	orbital_inclination	SP: This field is 0.0. SV: This field is 0.0. GS: This field is 0.0. CS:
DOUBLE	sat_ascending_node	SP: This field is 0.0. SV: This field is 0.0. GS: This field is 0.0. CS:
DOUBLE	sat_altitude	SP: This field is 0.0. SV: This field is 0.0. GS: This field is 0.0. CS:
DOUBLE	sat_ground_speed	SP: This field is 0.0. SV: This field is 0.0. GS: This field is 0.0. CS:

DOUBLE	satellite_heading	SP: This field is 0.0. SV: This field is 0.0. GS: This field is 0.0. CS:
DOUBLE	satellite_latitude	SP: This field is 0.0. SV: This field is 0.0. GS: This field is 0.0. CS:
DOUBLE	satellite_longitude	SP: This field is 0.0. SV: This field is 0.0. GS: This field is 0.0. CS:
DOUBLE	acrosstrackFOV	SP: This field is 0.0. SV: This field is 0.0. GS: This field is 0.0. CS:
DOUBLE	semi_major_axis	SP: This field is 0.0. SV: This field is 0.0. GS: This field is 0.0. CS:
DOUBLE[16]	bandgain	SP: These fields contain 0.0s. SV: These fields contain 0.0s. GS: These fields contain 0.0s. CS:
ENUM	LGSOWG	SP: This field is EMSC_FALSE. SV: This field is EMSC_FALSE. GS: This field is EMSC_FALSE. CS: This field is EMSC_TRUE.

TM Header

The following table defines the contents of the record written to an IMAGINE .img file which has been read

from a Landsat source. The fields are copied from the TM Fast Format headers. If the value is a string it is left unchanged. If the value is a number it is converted from ASCII to binary. All strings are NULL terminated. The description column indicates the source of the information. The term FF indicates the EOSAT Fast Format Header, and the term SF indicates EOSAT Standard Format Header, it is assumed that the user has access to the EOSAT documents.

SF: "Landsat Technical Working Group (LTWG)"

FF: "EOSAT Fast Format Document for TM Digital Products, Version B"

Type	Name	Description
LONG	imageheight	FF: Field 61. SF: table 8 bytes237−244.
LONG	imagewidth	FF: Field 59. SF: table 8 bytes 281−288
LONG	numbands	FF: based on Fields 94 and 35. SF: table 5 bytes 1413−1428.
CHAR[13]	eosatsceneid	FF: "NO SCENEID". SF: table 3 bytes 91−117
CHAR	rev	FF: Field 117. SF: blank.
CHAR[9]	acqdate	FF: Field 6. SF: blank.
CHAR[3]	satellitenum	FF: Field 8. SF: blank.
CHAR[5]	instrumenttype	FF: Field 10. SF: blank.
CHAR[12]	productnum	FF: Field 2. SF: blank.
CHAR[15]	producttype	FF: Field 12. SF: blank.
CHAR[11]	productsize	FF Field 14. SF: blank.

CHAR[77]	mapsheetname	FF: Field 15. SF: blank.
CHAR[4]	interleaving	FF: "BSQ". SF: table 3 bytes 131−154.
CHAR[5]	scenesize	FF: blank. SF: based on table 3 bytes 118−130.
CHAR[21]	earthellipsoid	FF: Field 51. SF: blank.
LONG	orbitalpath	FF: Field 4. SF: table 5 bytes 165−180.
LONG	orbitalrow	FF: Field 4. SF: table 5 bytes 165−180.
CHAR[5]	usgsprojection	FF: Field 43. SF: table 5 bytes 1557−1572.
LONG	usgsprojectionnum	FF: Field 45. SF: 0.
LONG	usgsmapzonenum	FF: Field 47. SF: 0.
DOUBLE	pixelsize	FF: Field 56. SF: table 6 bytes 365−380.
DOUBLE[15]	usgsprojparms	FF: Field 49. SF: 0.0s.
DOUBLE	upperleftlong	FF: Field 63. SF: 0.0.
DOUBLE	upperleftlat	FF: Field 65. SF: 0.0.
DOUBLE	upperlefteasting	FF: Field 67. SF: calculated from table 5 bytes 85−100, 101−106; pixelsize, scenesize.
DOUBLE	upperleftnorthing	FF: Field 69. SF: calculated from table 5 bytes 85−100, 101−106; pixelsize, scenesize.

DOUBLE	upperrightlong	FF: Field 71. SF: 0.0.
DOUBLE	upperrightlat	FF: Field 73. SF: 0.0.
DOUBLE	upperrighteasting	FF: Field 75. SF: calculated from table 5 bytes 85−100, 101−106; pixelsize, scenesize.
DOUBLE	upperrightnorthing	FF: Field 77. SF: calculated from table 5 bytes 85−100, 101−106; pixelsize, scenesize.
DOUBLE	lowerrightlong	FF: Field 79. SF: 0.0.
DOUBLE	lowerrightlat	FF: Field 81. SF: 0.0.
DOUBLE	lowerrighteasting	FF: Field 83. SF: calculated from table 5 bytes 85−100, 101−106; pixelsize, scenesize.
DOUBLE	lowerrightnorthing	FF: Field 85. SF: calculated from table 5 bytes 85−100, 101−106; pixelsize, scenesize.
DOUBLE	lowerleftlong	FF: Field 87. SF: 0.0.
DOUBLE	lowerleftlat	FF: Field 89. SF: 0.0.
DOUBLE	lowerlefteasting	FF: Field 91. SF: calculate from table 5 bytes 85−100, 101−106; pixelsize, scenesize.
DOUBLE	lowerleftnorthing	FF: Field 93. SF: calculated from table 5 bytes 85−100, 101−106; pixelsize, scenesize.
DOUBLE	scenecenterlat	FF: Field 107. SF: 0.0.

DOUBLE	scenecenterlong	FF: Field 105. SF: 0.0.
LONG	scenecenterline	FF: Field 113. SF: table 5 bytes 85−100.
LONG	scenecenterpixel	FF: Field 112. SF: table 5 bytes 101−106.
DOUBLE	scenecenternorthing	FF: Field 111. SF: table 6 141−156.
DOUBLE	scenecentereasting	FF: Field 109. SF: table 6 bytes 157−172.
CHAR[11]	geodproctype	FF: Field 17. SF: blank.
CHAR[3]	resampalgorithm	FF: Field 19. SF: table 5 bytes 1541−1556.
DOUBLE	semimajoraxis	FF: Field 53. SF: 0.0.
DOUBLE	semiminoraxis	FF: Field 55. SF: 0.0.
DOUBLE	radgainband1	FF: Field 21. SF: 0.0.
DOUBLE	radgainband2	FF: Field 23. SF: 0.0.
DOUBLE	radgainband3	FF: Field 25. SF: 0.0.
DOUBLE	radgainband4	FF: Field 27. SF: 0.0.
DOUBLE	radgainband5	FF: Field 29. SF: 0.0.
DOUBLE	radgainband6	FF: Field 31. SF: 0.0.

DOUBLE	radgainband7	FF: Field 33. SF: 0.0.
DOUBLE	radbiasband1	FF: Field 21. SF: 0.0.
DOUBLE	radbiasband2	FF: Field 23. SF: 0.0.
DOUBLE	radbiasband3	FF: Field 25. SF: 0.0.
DOUBLE	radbiasband4	FF: Field 27. SF: 0.0.
DOUBLE	radbiasband5	FF: Field 29. SF: 0.0.
DOUBLE	radbiasband6	FF: Field 31. SF: 0.0.
DOUBLE	radbiasband7	FF: Field 33. SF: 0.0.
CHAR[17]	wrsdesignator	FF: blank. SF:
CHAR[17]	wrscycle	FF: blank. SF:
LONG	sunelevation	FF: Field 101
LONG	sunazimuth	FF: Field 103
DOUBLE	orientationangle	FF: Field 41
DOUBLE	nominterlinedist	FF: 0.0. SF: table 6 bytes 61−76.
DOUBLE	nominterpixeldist	FF: 0.0. SF: table 6 bytes 45−60.
DOUBLE	nomaltitude	FF: 0.0. SF: table 6 bytes 493−508.
DOUBLE	nomgroundspeed	FF: 0.0. SF: table 6 bytes 509−524.

DOUBLE	crosstrackFOV	FF: 0.0. SF: table 6 bytes 557−572.
DOUBLE	displayrotangle	FF: 0.0. SF: table 6 bytes 445−460.
DOUBLE	scenecenterheading	FF: 0.0. SF: bytes 525−540.
DOUBLE	sensorscanrate	FF: 0.0. SF: table 6 bytes 573−588.
DOUBLE	sensorsamplerate	FF: 0.0. SF: table 6 bytes 589−604.
DOUBLE	nomorbitinclincation	FF: 0.0. SF: 0.0.
CHAR[6]	datum	FF: blank. SF: blank.
LONG	offset	FF: Field 115. SF: 0.0.