eixm

PACKAGE NAME

eixm – Raster Import/Export Package

SYNOPSIS

The Raster Import/Export Package (eixm) provides basic tools to access raster data external (non–IMAGINE) image formats. In addition, functions are provided to read ephemeris (header) data from IMAGINE files. Many IMAGINE importers write the ephemeris data from the source image to an emif structure stored as an ehfa node i IMAGINE file. The types of operations include:

eixm*HeaderRead –– Read the ephemeris data into a structure

eixm*HeaderWrite –– Write the ephemeris data into an IMAGINE file

eixm*HeaderFree –– Free the ephemeris data structure

DESCRIPTION

The Raster Import/Export Package (eixm) provides a set of common tools for impor from and exporting to non–IMAGINE file formats, and for accessing the ephemeris d an IMAGINE file placed there by an existing importer.

There are several variables which are common to most, if not all, raster importers a exporters. These include the width, height and number of bands in the source imag file or tape descriptor, etc. These variables are encapsulated into an *Eixm_Options* structure. A pointer to this structure is included in many calls to eixm functions.

DATA TYPES AND SYMBOLS

Macros

eixm_BinField:

Returns a binary data value of type *type*, beginning at byte number *offset* of *buffer*. useful for extracting binary information from a tape header record, for instance.

```
#define eixm_BinField(buf,offset,type) \
   *((type *)emsc_MemDup((buf)+(offset), sizeof(type)))
```

Enumerated Data Types

Eixm_DirMatch:

This is used to specify the behavior of the eixm_FindFiles function.

```
typedef enum {
      EIXM_INCLUDE_ALL = 0,  /* Find all files and directories */
      EIXM_EXCLUDE_DIRS,    /* Find files only; exclude directories */
      EIXM_DIRS_ONLY       /* Find directories only; exclude files */
} Eixm_DirMatch;
```

From ERDAS IMAGINE eixm On–Line Help. Copyright (c) 1982–1997 ERDAS, Inc.

Data Structures

Eixm_Options:

The *Eixm_Options* structure encapsulates the most commonly−used variables in import/export operations.

```
typedef struct eixm_options {

    /*
        ** Device options
        */
    Eixm_DeviceType      typ; /* Device type                          */
    char                *dev; /* Device name if typ isn't a file     */

    /*
        ** Attributes of the source
        */
    char                *ifn; /* Name of import source file          */

    int                 nob; /* Number of bands in input file       */
    long                ish; /* Input image size height             */
    long                isw; /* Input image size width              */
    double              mlrx; /* map/latlon lower right x coord      */
    double              mlry; /* map/latlon lower right y coord      */
    double              mulx; /* map/latlon upper left x coord       */
    double              muly; /* map/latlon upper left y coord       */
    Eixm_CoordinateType  mcrd; /* map coordinates                     */

    /*
        ** Output options
        */
    char                *ofn; /* Name of import destination file     */

    int                 ibd; /* Number of bands to import           */
    int                *bnd; /* Array of band numbers to import     */

    long                osh; /* Output image size height            */
    long                osw; /* Output image size width             */
    Emsc_Boolean        ulc; /* Upper left coordinates set          */
    double              ulx; /* Upper left X coordinate             */
    double              uly; /* Upper left Y coordinate             */
    Emsc_Boolean        lrc; /* Lower right coordinates set         */
    double              lrx; /* Lower right X coordinate            */
    double              lry; /* Lower right Y coordinate            */
    Eixm_CoordinateType  crd; /* Coordinate type (EIXM_FILE, etc.)   */

    Egda_PixelType      pix; /* Pixel type out(EDGA_TYPE_U8, etc)   */
    long                bsh; /* Blocksize height                    */
    long                bsw; /* Blocksize width                     */
    Edms_CompressionType  com; /* Obsolete - see compName field       */
    double              con; /* Contrast stretch multiplier         */
    Emsc_Boolean        sub; /* Create subsample pyramid            */
    Emsc_Boolean        zro; /* Ignore zeros in output stats        */
```

From ERDAS IMAGINE eixm On−Line Help. Copyright (c) 1982−1997 ERDAS, Inc.

```
    Emsc_Boolean        eph; /* Include Ephemeris data              */
    Emsc_Boolean        map; /* Include nominal map info            */

    /*
        ** Preview options
        */
    int             pre; /* Preview size (512 or 1024)          */
    int             red; /* Image reduction factor              */
    Eixm_DecimationType    dec; /* Decimation type (Average, etc.)     */

    /*
        ** Application-specific fields
        */
    Eixm_ApplicationType    app; /* Import application (TM, SPOT, etc) */
    Emsc_Opaque         *asp; /* Application-specific structure ptr */

    /*
        ** Miscellaneous pointers, flags, etc.
        */
    Emta_TapeId         tap; /* tape ID for tape devices            */
    Efio_Fd             fil; /* File ID for disk reads and writes   */
    int             fsk; /* Number of files to skip on media    */
    Emsc_Boolean        gui; /* EML interface is used               */
    long            rdl; /* record length for data              */
    Eerr_ErrorReportingMode rpt; /* Error reporting mode                */

    Evue_Connection     *apc; /* Connection to viewer process        */
    Evue_DisplayHandle    *vdh; /* Viewer display handle               */
    Evue_LayerHandle      vlh; /* Viewer layer handle                 */

    Eint_InitToolkitData   *erd; /* Pointer to programs erdinit         */
    Emet_MeterInfo      *mti; /* Meter info structure                */
    char            *mtr; /* Meter display name                  */
    struct eixm_options    *next; /* next in the linked list             */
    Eerr_ErrorReport      *err; /* Error pointer                       */
    Emsc_Boolean        comSet; /* Was com set on cmd line?            */
    Emsc_Boolean        subSet; /* Was sub set on cmd line?            */
    Emsc_Boolean        zroSet; /* Was zro set on cmd line?            */
    Eixm_CDROM          *cdrom; /* Info for CDROM reads and writes     */
    char            *pattern; /* Input (output) file pattern         */
    char            *compName; /* Compression name */
} Eixm_Options;
```

Enumerated Data Types

Eixm_CoordinateType:

Specifies the type of coordinates in which an image extent is stored.

```
typedef enum {
    EIXM_COORDINATE_NONE = 0, /* Not defined.
    EIXM_FILE,          /* UL and LR stored in file coordinates.  */
    EIXM_MAP,           /* UL and LR stored in map coordinates.   */
    EIXM_LONGLAT        /* UL and LR stored in long/lat coords.   */
} Eixm_CoordinateType;
```

From ERDAS IMAGINE eixm On−Line Help. Copyright (c) 1982−1997 ERDAS, Inc.

Eixm_DecimationType:

Specifies the type of decimation (sampling) which is done during an image reduc

```
typedef enum {
    EIXM_DECIMATE_NONE = 0,  /* Not defined.                                  */
    EIXM_NEARESTNEIGHBOR,    /* Nearest neighbor decimation.                  */
    EIXM_AVERAGEPIXELS,      /* Average of values in decimated area.    */
    EIXM_MAJORITYPIXEL       /* Most frequent value in decimated area. */
} Eixm_DecimationType;
```

Eixm_DeviceType:

Specifies the type of input (import) or output (export) device.

```
typedef enum {
    EIXM_DEVICE_NONE = 0,    /* Not defined.                                  */
    EIXM_DEVICE_FILE,        /* Source is a file.                          */
    EIXM_DEVICE_TAPE,        /* Source is a tape.                           */
    EIXM_DEVICE_CDROM        /* Source is a CDROM.                            */
} Eixm_DeviceType;
```

Data Structures

Eixm_Descrip:

The *Eixm_Descrip* structure contains all the information needed about the extern
i.e. the source (import) or destination (export) image. It is often used to describe
chunks of data which are written serially to the output device; e.g. strips, tiles, si
etc.

```
typedef struct _Eixm_Descrip {
    Eixm_DeviceType       deviceType;   /* File, tape, etc. */
    char                  *name;        /* File or device name */
    Emta_TapeId           tapeId;       /* Tape descriptor */
    Emta_TapeInfo         *tapeInfo;    /* Tape Information */
    int                   blockingFactor; /* Log. recs per phys. rec */
    Efio_Fd               fd;           /* File descriptor */
    Eixm_FormatType       format;       /* BIL, BIP, BSQ */
    unsigned short        numBands;     /* Number of bands in image */
    unsigned long         width;        /* Number of pixels wide */
    unsigned long         height;       /* Number of pixels high */
    Egda_PixelType        pixelType;    /* Unsigned 8 bit, etc. */
    Eixm_ByteOrder        byteOrder;    /* Intel, Motorola */
    unsigned long         recordLength; /* # bytes in logical record */
    unsigned long         lineHeaderBytes;/* # bytes skipped before */
                             /* image data starts */
    Eprj_MapInfo          *mapInfo;
    Eprj_ProParameters    *projection;
} Eixm_Descrip;
```

Eixm_InternalDescrip:

From ERDAS IMAGINE eixm On−Line Help. Copyright (c) 1982−1997 ERDAS, Inc.

The *Eixm_InternalDescrip* structure contains information about an IMAGINE file the destination (import) or source (export) image. It is a convenient way to enca information normally needed about an IMAGINE file for an import or export.

```
typedef struct _Eixm_InternalDescrip{
    Estr_StringList      *layerNames;  /* Names of IMAGINE layers  */
    Eimg_LayerStack      *layerstack;  /* Layerstack pointer       */
    long                 width;      /* Number of pixels wide    */
    long                 height;     /* Number of pixels high    */
    Eimg_LayerType       layerType;   /* Thematic or Athematic    */
    Egda_PixelType       pixelType;   /* Unsigned 8 bit, etc.     */
    long                 blockWidth;  /* Blocksize width          */
    long                 blockHeight; /* Blocksize width          */
    Edms_CompressionType compressType; /* Data Compression Type    */
    Emsc_Boolean         pyramidLayers; /* Create subsample pyramid */
    Emsc_Boolean         ignoreZero;  /* Ignore zeros in stats    */
    Eprj_MapInfo         *mapInfo;
    Eprj_ProParameters   *projection;
} Eixm_InternalDescrip;
```

Eixm_BandArray:

The *Eixm_BandArray* structure contains the number of bands to be imported or and an array of the band numbers to be processed.

```
typedef struct {
    int          count;     /* # of bands to process */
    int          *band;     /* The band array */
} Eixm_BandArray;
```

Eixm_Subset:

The *Eixm_Subset* structure contains the upper left and lower right coordinates o subset of the image to process, in file coordinates, as well as the subset width a and band subsetting information.

```
typedef struct {
    long                 ulx;  /* Upper left X file coordinate */
    long                 uly;  /* Upper left Y file coordinate */
    long                 lrx;  /* Lower right X file coordinate */
    long                 lry;  /* Lower right Y file coordinate */
    long                 width;     /* Subset width */
    long                 height;    /* Subset height */
    Eixm_BandArray       *bandArray;  /* # of bands to process */
} Eixm_Subset;
```

Enumerated Data Types

Eixm_ByteOrder:

Specifies the order of the bytes for multibyte data items.

From ERDAS IMAGINE eixm On−Line Help. Copyright (c) 1982−1997 ERDAS, Inc.

```
typedef enum {
    EIXM_MOTOROLA = 0,    /* High byte is first */
    EIXM_INTEL          /* Low byte is first  */
} Eixm_ByteOrder;
```

Data Structures

Eixm_Descrip:

The *Eixm_Descrip* structure contains all the information needed about the extern
i.e. the source (import) or destination (export) image. It is often used to describe
chunks of data which are written serially to the output device; e.g. strips, tiles, si
etc.

```
typedef struct _Eixm_Descrip {
    Eixm_DeviceType       deviceType;   /* File, tape, etc. */
    char                  *name;        /* File or device name */
    Emta_TapeId           tapeId;       /* Tape descriptor */
    Emta_TapeInfo         *tapeInfo;    /* Tape Information */
    int                   blockingFactor; /* Log. recs per phys. rec */
    Efio_Fd               fd;           /* File descriptor */
    Eixm_FormatType       format;       /* BIL, BIP, BSQ */
    unsigned short        numBands;     /* Number of bands in image */
    unsigned long         width;        /* Number of pixels wide */
    unsigned long         height;       /* Number of pixels high */
    Egda_PixelType        pixelType;    /* Unsigned 8 bit, etc. */
    Eixm_ByteOrder        byteOrder;    /* Intel, Motorola */
    unsigned long         recordLength; /* # bytes in logical record */
    unsigned long         lineHeaderBytes;/* # bytes skipped before */
                             /* image data starts */
    Eprj_MapInfo          *mapInfo;
    Eprj_ProParameters    *projection;
} Eixm_Descrip;
```

Data Structures

Eixm_InternalDescrip:

The *Eixm_InternalDescrip* structure contains information about an IMAGINE file
the destination (import) or source (export) image. It is a convenient way to enca
information normally needed about an IMAGINE file for an import or export.

```
typedef struct _Eixm_InternalDescrip{
    Estr_StringList       *layerNames;  /* Names of IMAGINE layers  */
    Eimg_LayerStack       *layerstack;  /* Layerstack pointer       */
    long                  width;        /* Number of pixels wide    */
    long                  height;       /* Number of pixels high    */
    Eimg_LayerType        layerType;    /* Thematic or Athematic    */
    Egda_PixelType        pixelType;    /* Unsigned 8 bit, etc.     */
    long                  blockWidth;   /* Blocksize width          */
    long                  blockHeight;  /* Blocksize width          */
    Edms_CompressionType  compressType; /* Data Compression Type    */
    Emsc_Boolean          pyramidLayers; /* Create subsample pyramid */
```

From ERDAS IMAGINE eixm On−Line Help. Copyright (c) 1982−1997 ERDAS, Inc.

```
    Emsc_Boolean        ignoreZero;  /* Ignore zeros in stats    */
    Eprj_MapInfo        *mapInfo;
    Eprj_ProParameters  *projection;
} Eixm_InternalDescrip;
```

Data Structures

Eixm_BandArray:

The *Eixm_BandArray* structure contains the number of bands to be imported or
and an array of the band numbers to be processed.

```
typedef struct {
    int         count;     /* # of bands to process */
    int         *band;     /* The band array */
} Eixm_BandArray;
```

Data Structures

Eixm_Subset:

The *Eixm_Subset* structure contains the upper left and lower right coordinates o
subset of the image to process, in file coordinates, as well as the subset width a
and band subsetting information.

```
typedef struct {
    long            ulx;  /* Upper left X file coordinate */
    long            uly;  /* Upper left Y file coordinate */
    long            lrx;  /* Lower right X file coordinate */
    long            lry;  /* Lower right Y file coordinate */
    long            width;     /* Subset width */
    long            height;    /* Subset height */
    Eixm_BandArray      *bandArray;   /* # of bands to process */
} Eixm_Subset;
```

Data Structures

Eixm_AdrgHeader:

The *Eixm_AdrgHeader* structure contains information from the header files from
ADRG source. It is stored as an ehfa node in an IMAGINE file.

```
typedef struct _Eixm_AdrgOutputHeader {
    EMIF_PTR(char) Originator; /* Name & address of originator */
    char  StockNum[17];      /* DMA stock number  */
    short VolEdNum;         /* Volume edition number */
    double PubDate;         /* Publication date */
    char SecurityClassification;/* Security Classification */
    Emsc_Boolean AgencyDeterminationReqd; /* Originating agency's
                                             determination required
    double DowngradeDate;       /* Date of downgrading */
    EMIF_PTR(char) Releasability;  /* Releaseability statement */
    EMIF_PTR(char) SpecIdent; /* Specification Identification for ADRG */
    double SpecDate;        /* Specification date */
```

From ERDAS IMAGINE eixm On−Line Help. Copyright (c) 1982−1997 ERDAS, Inc.

```
    char SpecAmendmentNum[21]; /* ADRG specification amendment number */
    char DistRectName[9];     /* Name of the DR directory */
    char ImageName[13];       /* Name of the image file */
    Eixm_AdrgZdrType ImageType; /* Type of image (overview, zdr, legend) */
    short Zone;           /* Zone number */
} Eixm_AdrgHeader;
```

Enumerated Data Types

Eixm_AdrgZdrType:

This specifies the type of image imported from an ADRG source: Zone Distributi Rectangle (ZDR) image, Overview image, or Legend image.

```
typedef enum {
    EIXM_ADRG_TYPE_UNKNOWN,
    EIXM_ADRG_ZDR,
    EIXM_ADRG_LEGEND,
    EIXM_ADRG_OVERVIEW
} Eixm_AdrgZdrType;
```

Data Structures

Eixm_AdriBandInfo:

The *Eixm_AdriBandInfo* structure contains information about the band(s) in an A image.

```
typedef struct _Eixm_AdriBandInfo {
    char BandColor[6];          /* Band color */
    long LowerEdgeWavelength;    /* Lower band edge wavelength (nm) */
    long UpperEdgeWavelength;    /* Upper band edge wavelength (nm) */
} Eixm_AdriBandInfo;
```

Eixm_AdriHeader:

The *Eixm_AdriHeader* structure contains information from the header files from t source. It is stored as an ehfa node in an IMAGINE file.

```
typedef struct _Eixm_AdriHeader {
    /*
        ** From the Volume Header
        */
    EMIF_PTR(char) Originator; /* Name & address of originator */
    char  StockNum[17];      /* DMA stock number  */
    short VolEdNum;         /* Volume edition number */
    double PubDate;         /* Publication date */
    char SecurityClassification;/* Security Classification */
    Emsc_Boolean AgencyDeterminationReqd; /* Originating agency's
                                             determination required
    double DowngradeDate;       /* Date of downgrading */
    EMIF_PTR(char) Releasability;  /* Releaseability statement */
    EMIF_PTR(char) SpecIdent; /* Specification Identification for ADRG */
    double SpecDate;        /* Specification date */
```

From ERDAS IMAGINE eixm On−Line Help. Copyright (c) 1982−1997 ERDAS, Inc.

```
    char SpecAmendmentNum[21]; /* ADRI specification amendment number */

    /*
         ** From the General Information File
         */
    Eixm_AdriImportType ImageType; /* Type of image (overview, zdr) */
    char DistRectName[9];     /* Name of the DR directory */
    char ImageName[13];       /* Name of the image file */
    short DataStructureType;  /* Number of Bands in original image */
    double DataDensityEW;       /* East-West Data Density */
    double DataDensityNS;       /* North-South Data Density */
    short DataDensityUnit;      /* Data Density Unit of Measure */
    double Ulx;         /* Latitude of UL corner of original image */
    double Uly;         /* Longitude of UL corner of original image */
    double Lrx;         /* Latitude of LR corner of original image */
    double Lry;         /* Longitude of LR corner of original image */
    double Scale;       /* Source image scale reciprocal */
    short Zone;           /* Zone number */
    double GSD; /* Ground Sample Distance (pixel spacing), nadir (m) */
    Emsc_Boolean Rectified; /* Image rectified during scanning? */
    double Asz;         /* ARC value Asz, adj. for source GSD & zone */
    double Bs;          /* ARC value Bs, adj. for source GSD */
    EMIF_PTR(char) Text;  /* Free text */
    EMIF_PTR(Eixm_AdriBandInfo) BandInfo; /* Band characteristics */

    /*
         ** From the Quality File
         */
    char EditionNumber[21];      /* Edition number of data set */
    double CreationDate;        /* Creation date of ZDR */
    double RevisionDate;        /* Date of revision or update */
    short RecompilationCount;    /* Recompilation count */
    short RevisionCount;        /* Revision count */
    EMIF_PTR(char) ADRIProduct;        /* Free text */
    double ProductSpecDate;  /* Date of ADRI product specification */
    char ProductSpecNum[21];     /* ADRI product specification number */
    double EarliestDate;  /* Earliest date of source image in DR */
    double LatestDate;    /* Latest date of source image in DR */
} Eixm_AdriHeader;
```

Enumerated Data Types

Eixm_AdriImportType:

This specifies the type of image imported from an ADRI source: Zone Distributio Rectangle (ZDR) image or Overview image.

```
typedef enum {
    EIXM_ADRI_IMPORT_OVERVIEW = 0,
    EIXM_ADRI_IMPORT_ZDR
} Eixm_AdriImportType;
```

Data Structures

Eixm_AvhrrHeader:

From ERDAS IMAGINE eixm On−Line Help. Copyright (c) 1982−1997 ERDAS, Inc.

The *Eixm_AvhrrHeader* structure contains information from the header files from source. It is stored as an ehfa node in an IMAGINE file.

```
typedef struct _Eixm_AvhrrHeader {
    /*
        ** TBM fields
        */
    Emif_Char      DataSetName[45];     /* Data set name */
    Emsc_Boolean   TotalCopy;           /* Total or selected copy? */
    Emsc_Boolean   LatitudeSelected;    /* Latitude select option used? */
    Emif_Short     BeginningLatitude;   /* Beginning latitude */
    Emif_Short     EndingLatitude;      /* Ending latitude */
    Emsc_Boolean   LongitudeSelected;   /* Long. select option used? */
    Emif_Short     BeginningLongitude;  /* Beginning longitude */
    Emif_Short     EndingLongitude;     /* Ending longitude */
    Emsc_Boolean   TimeSelected;        /* Time select option used? */
    Emif_Ushort    BeginningHour;       /* Beginning hour of data set */
    Emif_Ushort    BeginningMinute;     /* Beginning minute of data set */
    Emif_Ushort    NumMinutes;          /* Number of minutes contained */
    Emsc_Boolean   AppendedData;        /* Earth location data incl? */
    Emsc_Boolean   ChannelsSelected[20]; /* Is channel i selected? */

    /*
        ** Data Set Header fields
        */
    Emif_Char          SpacecraftID[10]; /* Name of the satellite */
    Eixm_AvhrrDataType     DataType;     /* GAC, LAC, etc. */
    Eixm_AvhrrTipSource    TipSource;    /* TIP source */
    Emif_Double        StartTime;        /* Time code from first frame of data proce
       for the data set */
    Emif_Ulong         NumScans;         /* Number of data scans */
    Emif_Double        EndTime;          /* Time code from last frame of data proce
       for the data set */
    Emif_Ulong         ProcessingBlockID; /* Field generated from start & end tim
    Emif_Uchar         CalibrationByte; /* Ramp/auto calibration field */
    Emif_Ushort        NumDataGaps;      /* Number of data gaps */
    Emif_Ushort        NoFrameSyncWordErrorsCount; /* Number of input data frames
       no frame sync word errors */
    Emif_Ushort        TipParityErrorsCount; /* Number of DACS detected TIP parit
       errors */
    Emif_Ushort        AuxiliarySyncErrorsCount; /* Sum of all auxiliary sync erro
       detected in input data */
    Emif_Uchar         CalibrationParameterID[2]; /* String of 2 8-bit characters *
    Emsc_Boolean       PseudoNoiseFlag; /* Normal or PseudoNoise data */
    Emif_Char          DataSource[10]; /* Receiving station */
    Eixm_AvhrrTapeDirection TapeDirection; /* Forward or reverse */
    Eixm_AvhrrDataMode     DataMode;     /* Test or flight data */
} Eixm_AvhrrHeader;
```

Enumerated Data Types

Eixm_AvhrrDataType:

This specifies the type of AVHRR data imported: LAC (Local Area Coverage), G

From ERDAS IMAGINE eixm On−Line Help. Copyright (c) 1982−1997 ERDAS, Inc.

(Global Area Coverage), HRPT (High Resolution Picture Transmission), and Du
format LAC data.

```
typedef enum {
    EIXM_AVHRR_DATATYPE_UNKNOWN = 0,
    EIXM_AVHRR_LAC,
    EIXM_AVHRR_GAC,
    EIXM_AVHRR_HRPT
    } Eixm_AvhrrDataType;
```

Eixm_AvhrrTipSource:

This specifies the Tiros Information Processor (TIP) source.

```
typedef enum {
    EIXM_AVHRR_TIPSOURCE_UNKNOWN = 0,
    EIXM_AVHRR_EMBEDDED_TIP,
    EIXM_AVHRR_STORED_TIP,
    EIXM_AVHRR_THIRD_CDA_TIP
    } Eixm_AvhrrTipSource;
```

Eixm_AvhrrTapeDirection:

```
typedef enum {
    EIXM_AVHRR_TAPE_DIR_UNKNOWN = 0,
    EIXM_AVHRR_REVERSE,       /* Time decrementing */
    EIXM_AVHRR_FORWARD        /* Time Incrementing */
    } Eixm_AvhrrTapeDirection;
```

Eixm_AvhrrDataMode:

```
typedef enum {
    EIXM_AVHRR_DATAMODE_UNKNOWN = 0,
    EIXM_AVHRR_TESTDATA,
    EIXM_AVHRR_FLIGHTDATA
    } Eixm_AvhrrDataMode;
```

Data Structures

Eixm_DemHeader:

The *Eixm_DemHeader* structure contains information from the header files from
source. It is stored as an ehfa node in an IMAGINE file.

```
typedef struct _Eixm_DemHeader {
    /*
        ** From the Type A Record
        */
    char   FileName[41];  /* DEM quadrangle name */
    char   FreeText[41];  /* Free format descriptor field */
    char   ProcessCode;   /* 1=GPM, 2=Manual Profile, 3=DLG2DEM, 4=DCLASS */
    char   SectionalIndicator[4]; /* Identifies 1:100000-scale sections */
    char   MCOriginCode[5];     /* Mapping center origin code */
    short  DEMLevelCode;  /* DEM Level */
```

From ERDAS IMAGINE eixm On−Line Help. Copyright (c) 1982−1997 ERDAS, Inc.

```
short  ElevationPatternCode;  /* 1=regular, 2=random */
short  GroundRefSystemCode;  /* 0=Geographic, 1=UTM, 2=State Plane */
short  Zone;  /* State Plane or UTM Zone */
double  ProjectionParams[15];  /* 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 */

/*
     ** ***************** Old Type A Format Stops Here ***************
     */
short  LargestPrimaryContourInterval;  /* Present only if 2 or more primary int
   exist */
short  LargestPrimaryIntervalUnits;  /* 0=N.A., 1=feet, 2=meters */
short  SmallestPrimaryContourInterval;  /* Smallest or only primary contour int
   */
short  SmallestPrimaryIntervalUnits;  /* 1=feet, 2=meters */
char  DataSourceDate[5];  /* YYMM format */
char  DataInspRevDate[5];  /* YYMM format */
char  InspectionRevisionFlag;  /* "I" or "R" */
short  DataValidationFlag;  /* See USGS manual */
short  SuspectVoidAreaFlag;  /* 0=none, 1=suspect areas, 2=void areas, 3=s
   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=Ol
   Hawaii Datum, 6=Puerto Rico Datum, 7=NAD 83 Provisional */
short  DataEdition;  /* Primarily DMA Specific field */

/*
     ** From the Type C (Accuracy) Record
     */
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 */
```

From ERDAS IMAGINE eixm On−Line Help. Copyright (c) 1982−1997 ERDAS, Inc.

```
    short   DemAccuracyZ;  /* In ElevationUnitCode units */
    long    DemSampleSize; /* 0=accuracy assumed to be estimated */
} Eixm_DemHeader;
```

Data Structures

Eixm_DoqHeader:

The *Eixm_DoqHeader* structure contains information from the header files from t
source. It is stored as an ehfa node in an IMAGINE file.

```
typedef struct _Eixm_DoqHeader {
    /*
        ** Record 1 -- Identifies the area of geographic coverage
        ** of the image
        */
    char quadrangleName[39];
    char quadrant[3];
    char nation1[3];
    char nation2[3];
    char state1[3];
    char state2[3];
    char state3[3];
    char state4[3];
    char state1county1[4];
    char state1county2[4];
    char state1county3[4];
    char state1county4[4];
    char state1county5[4];
    char state2county1[4];
    char state2county2[4];
    char state2county3[4];
    char state2county4[4];
    char state2county5[4];
    char state3county1[4];
    char state3county2[4];
    char state3county3[4];
    char state3county4[4];
    char state3county5[4];
    char state4county1[4];
    char state4county2[4];
    char state4county3[4];
    char state4county4[4];
    char state4county5[4];
    char descriptiveText[24];
    char producerCode[5];
    short dataOrdering;
    long numLines;      /* # lines in single band; i.e. height */
    long numSamples;    /* # samples in single band; i.e. width */
    short bandTypesAndOrder;
    short elevationStorage;
    short bandAndElevationStorage;
    short verticalDatum;
    short primaryHorizDatum;
    short secondaryHorizDatum;
```

From ERDAS IMAGINE eixm On−Line Help. Copyright (c) 1982−1997 ERDAS, Inc.

```
double rotationAngle;
short groundXYRefSystem;
short groundXYZone;
short groundXYUnits;
double SWPrimaryGroundCoord[2];
double NWPrimaryGroundCoord[2];
double NEPrimaryGroundCoord[2];
double SEPrimaryGroundCoord[2];

/*
     ** Record 2 -- Contains transformation parameters and coordinates
     ** to allow conversion from internal line and sample coordinates
     ** to the primary datum coordinate system
     */
double primaryCoeff[6];
double primaryCentroid[2];
double SWSecondaryGroundCoord[2];
double NWSecondaryGroundCoord[2];
double NESecondaryGroundCoord[2];
double SESecondaryGroundCoord[2];

/*
     ** Record 3 -- Contains transformation parameters and coordinates
     ** to allow conversion from internal line and sample coordinates
     ** to the secondary datum coordinate system
     */
double secondaryCoeff[6];
double secondaryCentroid[2];
long SWPrimaryInternalCoord[2];
long NWPrimaryInternalCoord[2];
long NEPrimaryInternalCoord[2];
long SEPrimaryInternalCoord[2];
long SWSecondaryInternalCoord[2];
long NWSecondaryInternalCoord[2];
long NESecondaryInternalCoord[2];
long SESecondaryInternalCoord[2];
double firstPixelPrimary[2];
double firstPixelSecondary[2];

/*
     ** Record 4 -- Contains information related to the source DEM
     ** and lineage information related to the digital orthophoto
     ** acquisition systems
     */
short elevationUnits;
float minElevation;
float maxElevation;
float groundResolutionX;
float groundResolutionY;
float groundResolutionZ;
float pixelResolutionX;
float pixelResolutionY;
float pixelResolutionZ;
long largestPrimaryContourInterval;
short largestPrimaryContourIntervalUnits;
long smallestPrimaryContourInterval;
```

From ERDAS IMAGINE eixm On−Line Help. Copyright (c) 1982−1997 ERDAS, Inc.

```
        short smallestPrimaryContourIntervalUnits;
        short suspectAndVoidAreas;
        float horizDoqAccuracy;
        float verticalDoqAccuracy;
        short numDoqHorizTestPoints;
        short pixelProcessingAlgorithm;
        char productionSystem[25];
        double productionDate;
        char filmType[25];
        char sourcePhotoID[25];
        short mosaickedImage;
        Emsc_Boolean leafOff;
        double sourcePhotoDate;
        float focalLength;
        long sourcePhotoFlyingHeight;
        char scannerType[25];
        float scanningResolution[2];
        float scannerSamplingResolution[2];
        short radiometricResolution;
        float resampledResolution;
} Eixm_DoqHeader;

extern Eixm_DoqHeader *eixm_DoqHeaderRead __((
        char *,
        Eint_InitToolkitData *,
        Eerr_ErrorReport **
));

extern void eixm_DoqHeaderWrite __((
        char *,
        Eixm_DoqHeader *,
        Eint_InitToolkitData *,
        Eerr_ErrorReport **
));

extern void eixm_DoqHeaderFree __((
        Eixm_DoqHeader *,
        Eerr_ErrorReport **
));
```

Data Structures

Eixm_DtedHeader:

The *Eixm_DtedHeader* structure contains information from the header files from DTED source. It is stored as an ehfa node in an IMAGINE file.

```
typedef struct _Eixm_DtedHeader {
    /*
        ** From the File Header Label (HDR)
        */
    char        FileName[18];       /* Reference filename */
    double      CreationDate;       /* Creation date of tape */
    double      ExpirationDate;     /* Expiration date of tape */
```

From ERDAS IMAGINE eixm On−Line Help. Copyright (c) 1982−1997 ERDAS, Inc.

```
/*
     ** From the User Header Label (UHL)
     */
char       SecurityCode[4];     /* T, S, C, U, or R */
char       UHL_ReferenceNumber[13];/* Unique reference number */
short      MultipleAccuracy;    /* 0=Single; 1=Muliple */

/*
     ** From the Data Set Identification (DSI) Record
     */
char       SecurityClassification; /* Security classification code */
char       SecurityHandlingDescription[28]; /* Other security description */
char       SeriesDesignator[6];  /* DMA Series Designator for product level
char       DSI_ReferenceNumber[16];/* Unique reference number */
short      DataEditionNumber;    /* Data Edition Number */
char       MatchMergeVersion;    /* Match/Merge Version */
char       MaintenanceDate[5];   /* Maintenance Date */
char       MatchMergeDate[5];    /* Match/Merge Date */
char       ProducerCode[9];     /* Producer Code */
char       ProductSpecStockNum[10];/* Product Specification Stock Number */
short      ProductSpecAmendmentNum;/* Product Specification Amendment and
   Change Number */
char       ProductSpecDate[5];  /* Product Specification Date */
char       VerticalDatum[4];    /* MSL (Mean Sea Level) */
char       HorizontalDatumCode[6]; /* WGS72 or WGS84 */
char       DigitizingCollectionSystem[11]; /* Free Text */
char       CompilationDate[5];   /* Compilation Date */

double      OriginLatitude;  /* Latitude of data origin */
double      OriginLongitude;  /* Longitude of data origin */
double      SWCornerLatitude; /* Latitude of SW data corner  */
double      SWCornerLongitude; /* Longitude of SW data corner */
double      NWCornerLatitude; /* Latitude of NW data corner  */
double      NWCornerLongitude; /* Longitude of NW data corner */
double      NECornerLatitude; /* Latitude of NE data corner  */
double      NECornerLongitude; /* Longitude of NE data corner */
double      SECornerLatitude; /* Latitude of SE data corner  */
double      SECornerLongitude; /* Longitude of SE data corner */
double      OrientationAngle; /* Clockwise orientatin angle of data with re
   to true North */
double      LatitudeInterval; /* Latitude interval in seconds */
double      LongitudeInterval; /* Longitude interval in seconds */
short      NumLatitudeLines; /* Number of rows containing data */
short      NumLongitudeLines; /* Number of columns containing data */
short      PartialCellIndicator; /* % of coverage completed (0=total) */

/*
     ** From the Accuracy Description (ACC) Record
     */
short      AbsoluteHorizontalAccuracy; /* Absolute horizontal accuracy in met
short      AbsoluteVerticalAccuracy;  /* Absolute vertical accuracy in meters
short      PointToPointHorizAcc;    /* Point-to-Point horizontal accuracy in
   meters */
short      PointToPointVertAcc;     /* Point-to-Point vertical accuracy in me
   */
```

From ERDAS IMAGINE eixm On−Line Help. Copyright (c) 1982−1997 ERDAS, Inc.

```
} Eixm_DtedHeader;
```

Data Structures

Eixm_MssTickMark:

The *Eixm_MssTickMark* structure contains the tick mark information stored in th header.

```
typedef struct _Eixm_MssTickMark {
    unsigned short position;     /* Row or column position */
    char label[8];           /* Tick label */
} Eixm_MssTickMark;
```

Eixm_MssHeader:

The *Eixm_MssHeader* structure contains information from the header files from source. It is stored as an ehfa node in an IMAGINE file.

```
typedef struct _Eixm_MssHeader {

    /*
        ** From the tape directory
        */
    char  MissionNum;         /* Landsat 1, 2 or 3 */
    double TapeCreationDate;     /* Date of CCT Tape Creation */
    unsigned short NumVolumes;    /* Number of volumes in CCT set */
    Eixm_FormatType format;      /* BIL or BSQ */
    unsigned short RecordLength;  /* 3596 bytes */
    Emsc_Boolean SourceHDT;      /* C or U (Corrected or Uncorrected data) */
    char SceneID[19];         /* Unique scene identifier */

    /*
        ** From the Header Record
        */
    char WRSDesignator[8];       /* Node direction, path and row */
    char SensorID[4];          /* MSS */
    unsigned short OrbitNum;     /* Orbit number of the spacecraft */
    unsigned short ActiveDetectorStatus[26]; /* 1=Active, 0=Inactive */
    unsigned short ActiveDetectorCount;    /* Number of active detectors */
    unsigned short UncorrectedPixelsPerLine; /* Pixels per scan line in original
       geometrically uncorrected image */

    unsigned short WRSScanLineNum; /* Scan line number of WRS (World Referenc
       System) */
    unsigned short WRSPixelNum;   /* Pixel number of WRS */

    unsigned short NumAnnotationRecords; /* 1 if corrected, 2 if uncorr. */
    unsigned short NumAncillaryRecords; /* 0 if corrected, 26 if uncorr. */

    Emsc_Boolean GeoCorrectionsApplied; /* Geometric corrections applied */
    Emsc_Boolean GeoCorrectionDataPresent; /* Correction data present */
    Emsc_Boolean RadiometricCorrectionsApplied; /* Radiometric correction applied
    Emsc_Boolean RadiometricCorrectionDataPresent; /* Radiometric correction data
       present */
```

From ERDAS IMAGINE eixm On−Line Help. Copyright (c) 1982−1997 ERDAS, Inc.

```
Eixm_MssImageFormat ImageDataFormat; /* Framed or unframed, square or
   rectangular */
unsigned short LineInterleavingCount; /* 0 for BSQ or 255 for BIL */
unsigned short BitsPerPixel;       /* Always 8 */
Eixm_MssResamplingMethod ResamplingApplied; /* None, Cubic Convoluation, or
   Nearest Neighbor */
short WRSImageCenterOffset; /* WRS offset from fully processed image cente
unsigned short PixelsPerScanLine; /* Number of pixels per scan line */
unsigned short NumUsableImages;  /* Number of usable images (subscenes) pe
   */

unsigned short NumTrailerRecords; /* Nominally one per image */
Emsc_Boolean DayPass;        /* True = Day, False = Night */
Eixm_MssCalWedgeMode CalWedgeMode; /* Cal Wedge Mode */

/*
     ** Temporal Registration Data
     */
unsigned short TempRegPt1CurImgScanLine;
unsigned short TempRegPt1CurImgPixelNum;
unsigned short TempRegPt1RefImgScanLine;
unsigned short TempRegPt1RefImgPixelNum;
unsigned short TempRegPt2CurImgScanLine;
unsigned short TempRegPt2CurImgPixelNum;
unsigned short TempRegPt2RefImgScanLine;
unsigned short TempRegPt2RefImgPixelNum;
unsigned short TempRegPt3CurImgScanLine;
unsigned short TempRegPt3CurImgPixelNum;
unsigned short TempRegPt3RefImgScanLine;
unsigned short TempRegPt3RefImgPixelNum;
unsigned short TempRegPt4CurImgScanLine;
unsigned short TempRegPt4CurImgPixelNum;
unsigned short TempRegPt4RefImgScanLine;
unsigned short TempRegPt4RefImgPixelNum;

/*
     ** Overlap Data
     */
unsigned short OverlapMark1ScanLine;
unsigned short OverlapMark1PixelNum;
unsigned short OverlapMark2ScanLine;
unsigned short OverlapMark2PixelNum;
unsigned short OverlapMark3ScanLine;
unsigned short OverlapMark3PixelNum;
unsigned short OverlapMark4ScanLine;
unsigned short OverlapMark4PixelNum;
unsigned short OverlapMarkPixelOffset;

unsigned short GeoModelQualityAssessment; /* 9=highest, 0=lowest */
unsigned short NumTickMarksTop;  /* Number of tick marks, top */
unsigned short NumTickMarksLeft;  /* Number of tick marks, left */
unsigned short NumTickMarksRight; /* Number of tick marks, right */
unsigned short NumTickMarksBottom; /* Number of tick marks, bottom */

Emsc_Boolean EdipsContrastEnhancement; /* EDIPS performed contrast enhance
```

From ERDAS IMAGINE eixm On−Line Help. Copyright (c) 1982−1997 ERDAS, Inc.

```
    Emsc_Boolean EdipsAtmScatterCompensation; /* EDIPS performed atmospheric sc
       compensation */
    Emsc_Boolean EdipsEdgeEnhancement; /* Edips performed edge enhancement */
    unsigned short DataPresentByBand[8]; /* 1=data present for band */

    /*
         ** From the Annotation Record
         */
    float FormatCenterLatitude;   /* Latitude at the center of the MSS image for
    float FormatCenterLongitude;  /* Longitude at the center of the MSS image f
       */
    char NominalPathRowID[10];    /* Nominal path row identifier */
    float NominalLatitude;       /* Nominal latitude */
    float NominalLongitude;      /* Nominal longitude */
    char SpectralBandIDCode[8];   /* Sensor spectral-band identification code *
    Eixm_MssTransmissionType TransmissionType; /* Direct or stored */
    char SunAngles[15]; /* Elevation and azimuth angle measured clockwise fro
       North, nearest degree */
    Eixm_MssCorrectionType CorrectionType; /* Type of correction applied */
    char ImageScale[16];             /* Scale of image */
    Eixm_MssMapProjection MapProjection;  /* Map projection */
    Eixm_MssEphemerisType EphemerisType;  /* Type of ephemeris data used to c
       image center */
    Eixm_MssProcessingProcedure ProcessingProcedure; /* Normal or abnormal */
    Eixm_MssSensorGain SensorGain; /* High or low gain */
    char AgencyProject[14];      /* NASA Landsat */
    char FrameID[16];           /* Unique identifier for image */
    Eixm_MssTickMark TopTickMark[16];     /* Top tick mark data */
    Eixm_MssTickMark LeftTickMark[25];    /* Left tick mark data */
    Eixm_MssTickMark RightTickMark[25];   /* Right tick mark data */
    Eixm_MssTickMark BottomTickMark[16];  /* Bottom tick mark data */
} Eixm_MssHeader;
```

Enumerated Data Types

Eixm_MssImageFormat

```
typedef enum {
    EIXM_MSS_IMAGE_FORMAT_UNKNOWN = 0,
    EIXM_MSS_UNFRAMED_RECTANGULAR_IMAGE,  /* Unframed rectangular */
    EIXM_MSS_FRAMED_RECTANGULAR_IMAGE,    /* Framed rectangular */
    EIXM_MSS_FRAMED_SQUARE_IMAGE        /* Framed square */
} Eixm_MssImageFormat;
```

Eixm_MssResamplingMethod:

```
typedef enum {
    EIXM_MSS_RESAMPLING_NONE = 0,  /* No resampling */
    EIXM_MSS_CUBIC_CONVOLUTION,    /* Cubic Convolution resampling */
    EIXM_MSS_NEAREST_NEIGHBOR     /* Nearest Neighbor resampling */
} Eixm_MssResamplingMethod;
```

Eixm_MssMapProjection:

```
typedef enum {
```

From ERDAS IMAGINE eixm On−Line Help. Copyright (c) 1982−1997 ERDAS, Inc.

```
    EIXM_MSS_MAP_PROJECTION_NONE = 0,    /* No map projection */
    EIXM_MSS_UTM,                /* UTM */
    EIXM_MSS_POLAR_STEREOGRAPHIC,       /* Polar Stereographic */
    EIXM_MSS_HOTINE_OBLIQUE_MERCATOR,    /* Hotine Oblique Mercator */
    EIXM_MSS_SPACE_OBLIQUE_MERCATOR,     /* Space Oblique Mercator */
    EIXM_MSS_LAMBERT             /* Lambert */
} Eixm_MssMapProjection;
```

Eixm_MssCalWedgeMode:

```
typedef enum {
    EIXM_MSS_CAL_WEDGE_MODE_UNKNOWN = 0,
    EIXM_MSS_LOW_GAIN_LINEAR_TRANSMISSION,
    EIXM_MSS_LOW_GAIN_COMPRESSED_TRANSMISSION,
    EIXM_MSS_HIGH_GAIN_LINEAR_TRANSMISSION,
    EIXM_MSS_HIGH_GAIN_COMPRESSED_TRANSMISSION
} Eixm_MssCalWedgeMode;
```

Eixm_MssTransmissionType:

```
typedef enum {
    EIXM_MSS_TRANSMISSION_UNKNOWN = 0,
    EIXM_MSS_DIRECT,             /* Direct transmission */
    EIXM_MSS_STORED_PLAYBACK         /* Stored playback */
} Eixm_MssTransmissionType;
```

Eixm_MssCorrectionType:

```
typedef enum {
    EIXM_MSS_CORRECTION_NONE = 0,
    EIXM_MSS_SYSTEM_CORRECTION,
    EIXM_MSS_GEOMETRIC_GCP_CORRECTION,
    EIXM_MSS_RELATIVE_GCP_CORRECTION
} Eixm_MssCorrectionType;
```

Eixm_MssEphemerisType:

```
typedef enum {
    EIXM_MSS_EPHEMERIS_TYPE_UNKNOWN = 0,
    EIXM_MSS_PREDICTIVE,
    EIXM_MSS_DEFINITIVE
} Eixm_MssEphemerisType;
```

Eixm_MssProcessingProcedure:

```
typedef enum {
    EIXM_MSS_PROCESSING_PROCEDURE_UNKNOWN = 0,
    EIXM_MSS_ABNORMAL,
    EIXM_MSS_NORMAL
} Eixm_MssProcessingProcedure;
```

Eixm_MssSensorGain:

```
typedef enum {
```

From ERDAS IMAGINE eixm On−Line Help. Copyright (c) 1982−1997 ERDAS, Inc.

```
    EIXM_MSS_SENSOR_GAIN_UNKNOWN = 0,
    EIXM_MSS_HIGH_GAIN,
    EIXM_MSS_LOW_GAIN
} Eixm_MssSensorGain;
```

Data Structures

Eixm_SpotHeader:

The *Eixm_SpotHeader* structure contains information from the header files from source. It is stored as an ehfa node in an IMAGINE file.

```
typedef struct _eixm_SpotHeader {
    char   scnname[15];  /* constructed by the packed DMS scene  */
                         /* center coordinates and the prefix    */
                         /* 'SP'                                 */

    double scncenlat;    /* scene center latitude in decimal degrees  */
    double scncenlon;    /* scene center longitude in decimal degrees */
    double scncor1lat;   /* upper left latitude in decimal degrees    */
    double scncor1lon;   /* upper left longitude in decimal degrees   */
    double scncor2lat;   /* lower left latitude in decimal degrees    */
    double scncor2lon;   /* lower left longitude in decimal degrees   */
    double scncor3lat;   /* upper right latitude in decimal degrees   */
    double scncor3lon;   /* upper right longitude in decimal degrees  */
    double scncor4lat;   /* lower right latitude in decimal degrees   */
    double scncor4lon;   /* lower right longitude in decimal degrees  */
    long   scncenline;   /* scene center line number                  */
    long   scncenpixel;  /* scene center pixel(column) number         */
    long   scncor1line;  /* upper left line number                    */
    long   scncor1pixel; /* upper left pixel(column) number           */
    long   scncor2line;  /* lower left line number                    */
    long   scncor2pixel; /* lower left pixel(column) number           */
    long   scncor3line;  /* upper right line number                   */
    long   scncor3pixel; /* upper right pixel(column) number          */
    long   scncor4line;  /* lower right line number                   */
    long   scncor4pixel; /* lower right pixel(column) number          */

    double scnorientangle; /* scene orientation angle                 */
    double angleofinc;   /* angle of incidence
                            ** (negative to the west of scene center,
                            ** (positivie to the east of scene cente
                            */
    double sunazimuth;   /* sun azimuth in decimal degrees            */
    double sunelevation; /* sun elevation                             */

    char   scncentime[17]; /* date in the form of hh:mm mm/dd/yy      */

    char   sensorid[4];  /* sensor identification                     */
    char   specmode[3];  /* spectral mode PAN or XS                   */
    long   revolutionnum; /* revolution number within cycle           */
    char   playback[2];  /* direct playback indicator DT,E1,E2        */

    char       preprocesslevel[9]; /* preprocess level 1A, 1B, 2      */
    Emsc_Boolean correction;       /* radiometric correction applied */
```

From ERDAS IMAGINE eixm On−Line Help. Copyright (c) 1982−1997 ERDAS, Inc.

```
long      deconvolution;    /* 0 for 1A; 1 for all others     */
char      resampling;      /* resampling designator  */

long   ulxmap;      /* upper left x map coordinate */
long   ulymap;      /* upper left y map coordinate */
long   lrxmap;      /* lower right x map coordinate */
long   lrymap;      /* lower right y map coordinate */
long   xpixelsize;   /* pixel size along x-axis 10=PAN; 20=XS     */
long   ypixelsize;   /* pixel size along y-axis 10=PAN; 20=XS     */
double mapxpixelsize; /* map projection image size (x-axis)        */
double mapypixelsize; /* map projection image size (y-axis)        */
char   mapprojid[32]; /* map projection identifier (level 2 only)  */

long   numcols;     /* image width  */
long   numlines;    /* image height */
long   numbands;     /* number of bands */

char   interleave[4]; /* image interleaving factor */

char   cartcoord[33]; /* cartographic coordinates of  first pixel*/
              /* in first line                              */
char   suncal[16];   /* sun calibration date                    */
char   mapunits[8];   /* map units                               */
char   mapproj[35];   /* map projection                          */
long   utmzone;     /* utm zone number                         */
long   utmeast;     /* UTM upper left easting                  */
long   utmnorth;    /* UTM upper left northing                 */
long   stplanecode;   /* NOS code                                */
long   stplaneeast;   /* State Plane upper left easting          */
long   stplanenorth;  /* State Plane upper left northing         */
char   hdatum[6];    /* Horizontal datum                        */
```

Data Structures

Eixm_TMHeader:

The *Eixm_TMHeader* structure contains information from the header files from tl
source. It is stored as an ehfa node in an IMAGINE file.

```
typedef struct {
    /*
         ** Specific information about image.
         */
    long   imageheight;       /* number of rows               */
    long   imagewidth;        /* number of columns            */
    long   numbands;          /* number of bands in image     */

    /*
         ** Specific information about physical tape.
         */
    char   eosatsceneid[13];    /* eosat scene id (std. format) */
    char   rev;            /* FF Rev A or B (RevB)          */
    char   acqdate[9];        /* acquisition date             */
    char   satellitenum[3];     /* satellite number             */
    char   instrumenttype[5];   /* instrument type FF RevB      */
```

From ERDAS IMAGINE eixm On–Line Help. Copyright (c) 1982–1997 ERDAS, Inc.

```
char   productnum[12];     /* product order number FF RevB */
char   producttype[15];    /* product type FF RevB         */
char   productsize[11];    /* product size FF RevB         */
char   mapsheetname[77];   /* map sheet name FF RevB       */
char   interleaving[4];    /* BIL or BSQ                   */
char   scenesize[5];       /* FULL or QUAD                 */

/*
     ** Georgraphic information about image.
     */
char   earthellipsoid[21];  /* ellipsoid used              */
long   orbitalpath;        /* orbital path of satellite    */
long   orbitalrow;         /* orbital row of satellite     */

char   usgsprojection[5];   /* USGS projection name        */
long   usgsprojectionnum;   /* USGS projection number      */
long   usgsmapzonenum;      /* USGS map zone number        */
double pixelsize;          /* pixel size                   */

double usgsprojparms[15];

double upperleftlong;      /* upper left longitude of image */
double upperleftlat;       /* upper left latitude of image  */
double upperlefteasting;   /* upper left easting in meters  */
double upperleftnorthing;  /* upper left northing in meters */
double upperrightlong;     /* upper right longitude of image*/
double upperrightlat;      /* upper right latitude of image */
double upperrighteasting;  /* upper right easting in meters */
double upperrightnorthing; /* upper right northing in meters*/
double lowerrightlong;     /* lower right longitude of image*/
double lowerrightlat;      /* lower right latitude of image */
double lowerrighteasting;  /* lower right easting in meters */
double lowerrightnorthing; /* lower right northing in meters*/
double lowerleftlong;      /* lower left longitude of image */
double lowerleftlat;       /* lower left latitude of image  */
double lowerlefteasting;   /* lower left easting in meters  */
double lowerleftnorthing;  /* lower left northing in meters */

double scenecenterlat;     /* scene center latitude of image */
double scenecenterlong;    /* scene center longitude of image*/
long   scenecenterline;    /* scene center line number       */
long   scenecenterpixel;   /* scene center pixel number      */
double scenecenternorthing; /* scene center northing in meters*/
double scenecentereasting;  /* scene center easting in meters */

char   geodproctype[11];   /* type of geodetic processing   */
char   resampalgorithm[3];  /* type of resampling algorithm  */

double semimajoraxis;      /* semi major axis of earth     */
double semiminoraxis;      /* semi minor axis of earth     */

double radgainband1;       /* radiance gain for band 1 RevB */
double radgainband2;       /* radiance gain for band 2 RevB */
double radgainband3;       /* radiance gain for band 3 Rev */
double radgainband4;       /* radiance gain for band 4 RevB */
```

From ERDAS IMAGINE eixm On−Line Help. Copyright (c) 1982−1997 ERDAS, Inc.

```
    double  radgainband5;      /* radiance gain for band 5 RevB */
    double  radgainband6;      /* radiance gain for band 6 RevB */
    double  radgainband7;      /* radiance gain for band 7 RevB */

    double  radbiasband1;      /* radiance bias for band 1 RevB */
    double  radbiasband2;      /* radiance bias for band 2 RevB */
    double  radbiasband3;      /* radiance bias for band 3 RevB */
    double  radbiasband4;      /* radiance bias for band 4 RevB */
    double  radbiasband5;      /* radiance bias for band 5 RevB */
    double  radbiasband6;      /* radiance bias for band 6 RevB */
    double  radbiasband7;      /* radiance bias for band 7 RevB */

    char    wrsdesignator[17];   /* WRS designator RevB          */
    char    wrscycle[17];       /* WRS cycle                    */

    /*
         ** Other ephemeris data
         */
    long    sunelevation;       /* sun elevation in degrees     */
    long    sunazimuth;         /* sun azimuth in degrees       */
    double  orientationangle;    /* orientation angle in degrees */

    /*
          * Newly added pieces of data
          */
    double nominterlinedist;  /* nominal interline distance */
    double nominterpixeldist; /* nominal interpixel distance */

    double nomaltitude;     /* nominal altitude in KM */
    double nomgroundspeed;   /* nominal ground speed in KM/sec */
    double crosstrackFOV;    /* cross track FOV in degrees */

    double displayrotangle;  /* display rotation angle */
    double scenecenterheading; /* full scene center heading */
    double sensorscanrate;   /* sensor scan rate/sec */
    double sensorsamplerate;  /* sensor active sampling rate/sec */
    double nomorbitinclination;/* satellite orbit inclination */

    char datum[6];         /* UTM datum */
    long offset;          /*
                                    ** horizontal offset of true scene cer
                                    ** assuming WRS coordinate system
                                    */
} Eixm_TMHeader;
```

From ERDAS IMAGINE eixm On−Line Help. Copyright (c) 1982−1997 ERDAS, Inc.