

File created: 11-Nov-88 14:17:22 {ERINYES}<LISPUSERS>LYRIC>AISBLT.;1

previous date: 23-Sep-88 20:28:25 {PHYLUM}<LISPUSERS>MEDLEY>AISBLT.;1

Read Table: XCL

Package: INTERLISP

Format: XCCS

; Copyright (c) 1988 by Xerox Corporation. All rights reserved.

(RPAQQ **AISBLTCOMS**

(,;

;; The AISBLT module

;;

;; The exported interface is via the following functions

(FNS AISBLT.BITMAP AISFILEHEADER WRITEAIS)

;; Internal functions

(FNS AISBLT1TO1.BITMAP AISBLT8TO1FSA.BITMAP AISBLT8TO8.BITMAP AISBLTNT01FSA.BITMAP  
AISBLTNT01TRUNCATE.BITMAP SETUPPILOTBBT)

;; Useful constants

(COMS ; AIS file identifying word

(VARIABLES AIS-PASSWORD))

(COMS ; Attribute part header types

(VARIABLES AIS-RASTER-TYPE AIS-PLACEMENT-TYPE AIS-PHOTOMETRY-TYPE AIS-COMMENT-TYPE))

(COMS ; Raster coding types

(VARIABLES AIS-RASTER-CODING-UCA AIS-RASTER-CODING-CA))

(COMS ; Photometry sense

(VARIABLES AIS-PHOTOMETRY-SENSE-LARGER-DARKER AIS-PHOTOMETRY-SENSE-LARGER-LIGHTER))

(COMS ; Photometry signal

(VARIABLES AIS-PHOTOMETRY-SIGNAL-BLACK-AND-WHITE AIS-PHOTOMETRY-SIGNAL-RED-SEPARATION  
AIS-PHOTOMETRY-SIGNAL-BLUE-SEPARATION AIS-PHOTOMETRY-SIGNAL-GREEN-SEPARATION  
AIS-PHOTOMETRY-SIGNAL-CYAN-SEPARATION AIS-PHOTOMETRY-SIGNAL-MAGENTA-SEPARATION  
AIS-PHOTOMETRY-SIGNAL-YELLOW-SEPARATION AIS-PHOTOMETRY-SIGNAL-X-CIE  
AIS-PHOTOMETRY-SIGNAL-Y-CIE AIS-PHOTOMETRY-SIGNAL-IN-COMMENT  
AIS-PHOTOMETRY-SIGNAL-UNSPECIFIED))

(COMS ; Photometry spot type

(VARIABLES AIS-PHOTOMETRY-SPOT-TYPE-RECTANGULAR AIS-PHOTOMETRY-SPOT-TYPE-CIRCULAR  
AIS-PHOTOMETRY-SPOT-TYPE-IN-COMMENTS AIS-PHOTOMETRY-SPOT-TYPE-UNSPECIFIED))

(COMS ; Photometry scale

(VARIABLES AIS-PHOTOMETRY-SCALE-RELECTANCE-TRANSMITTANCE AIS-PHOTOMETRY-SCALE-OPTICAL-DENSITY  
AIS-PHOTOMETRY-SCALE-IN-COMMENT AIS-PHOTOMETRY-SCALE-UNSPECIFIED))

(COMS ; Photometry

(VARIABLES AIS-PHOTOMETRY-UNSPECIFIED))

(COMS ; Header information when writing AIS format

(VARIABLES AIS-DEFAULT-HEADER-LENGTH)))

;;

;; The AISBLT module

;;

;; The exported interface is via the following functions

(DEFINEQ

(**AISBLT.BITMAP**

(LAMBDA (FILE SOURCE-LEFT SOURCE-BOTTOM DESTINATION DESTINATION-LEFT DESTINATION-BOTTOM WIDTH HEIGHT HOW  
FILTER)

; Edited 23-Sep-88 19:48 by Briggs

; Edited 2-May-88 16:51 by Briggs

; Edited 2-May-88 16:06 by Briggs

;;; Puts an AIS image from a file onto the destination, which may be a bitmap, or a window/imagestream.

;;; The arguments are the same as BITBLTs for the most part.

;; HOW specifies how the number of bits per pixel is condensed if reduction is necessary: TRUNCATE is truncate; FSA is Floyd-Steinberg  
;; algorithm; MODULATE is modulated with a random function

(RESETLST

(PROG (AIS-HEADER RASTER-PART PHOTOMETRY-PART PHOTOMETRY-SENSE CLIP-BOTTOM CLIP-HEIGHT CLIP-LEFT  
CLIP-RIGHT CLIP-TOP CLIP-WIDTH DESTINATION-BASE DESTINATION-BITS-PER-PIXEL  
DESTINATION-RASTER-WIDTH DESTINATION-WIDTH S-TO-D-X S-TO-D-Y SCAN-DIRECTION  
SOURCE-BITS-PER-PIXEL SOURCE-HEIGHT SOURCE-PIXEL-OFFSET SOURCE-RASTER-WIDTH SOURCE-WIDTH  
STREAM)

;; check and default some of the parameters

(OR (TYPEP DESTINATION 'BITMAP)

(\ILLEGAL.ARG DESTINATION))

(OR SOURCE-LEFT (SETQ SOURCE-LEFT 0))

(OR SOURCE-BOTTOM (SETQ SOURCE-BOTTOM 0))

(OR DESTINATION-LEFT (SETQ DESTINATION-LEFT 0))

(OR DESTINATION-BOTTOM (SETQ DESTINATION-BOTTOM 0))

```

(OR HOW (SETQ HOW 'FSA))
;; find the file, and get the AIS image parameters
(COND
  ((STREAMP FILE)
   (SETQ STREAM FILE))
  ((SETQ STREAM (FINDFILE FILE T AISDIRECTORIES))
   (RESETSAVE NIL (LIST 'CLOSEF (SETQ STREAM (OPENSTREAM STREAM 'INPUT NIL '(SEQUENTIAL T))))))
  (T (ERROR "Can't find file" FILE)))
;; interesting point here -- INSUREAISFILE should probably also check for Photometry information and indicate whether the sense of the
;; samples is increasing values implies increasing lightness or the other way around. Currently, for the 1 bpp case we invert the bitmap so
;; that it displays correctly on the screen.
(SETQ AIS-HEADER (AISFILEHEADER STREAM))
(SETQ RASTER-PART (CADR (ASSOC :RASTER AIS-HEADER)))
;; dispose of some of the cases we can't handle
(COND
  ((NOT (EQ (LISTGET RASTER-PART :CODING-TYPE)
            AIS-RASTER-CODING-UCA))
   (ERROR "Can't AISBLT AIS files of raster coding type" (LISTGET RASTER-PART :CODING-TYPE))))
;; extract the information we need from the raster attribute
(SETQ SOURCE-BITS-PER-PIXEL (LISTGET RASTER-PART :BITS-PER-SAMPLE))
(SETQ SOURCE-WIDTH (LISTGET RASTER-PART :SCAN-LENGTH))
(SETQ SOURCE-HEIGHT (LISTGET RASTER-PART :SCAN-COUNT))
(SETQ SOURCE-RASTER-WIDTH (LISTGET RASTER-PART :WORDS-PER-SCAN-LINE))
(SETQ SCAN-DIRECTION (LISTGET RASTER-PART :SCAN-DIRECTION))
;; Dispose of another case we don't want to handle right now
(COND
  ((NOT (EQ SCAN-DIRECTION 3))
   (ERROR "Scan direction is not top-left to bottom-right(3) - " SCAN-DIRECTION)))
;; extract the information we need from the photometry part
(SETQ PHOTOMETRY-PART (CADR (ASSOC :PHOTOMETRY AIS-HEADER)))
;; the photometry sense will indicate whether we need to invert the bitmap to get it into Lisp's 0->white 1-> black sense (larger darker).
(SETQ PHOTOMETRY-SENSE (OR (LISTGET PHOTOMETRY-PART :SENSE)
                           (COND
                            ((EQ SOURCE-BITS-PER-PIXEL 0)
                             ;; this is a gross kludge by Cedar to avoid specifying the photometry information
                             (SETQ SOURCE-BITS-PER-PIXEL 1)
                             AIS-PHOTOMETRY-SENSE-LARGER-DARKER)
                            (T AIS-PHOTOMETRY-SENSE-LARGER-LIGHTER))))))
;; calculate some additional destination information
(SETQ DESTINATION-WIDTH (|fetch| (BITMAP BITMAPWIDTH) |of| DESTINATION))
(SETQ DESTINATION-RASTER-WIDTH (|fetch| (BITMAP BITMAPRASTERWIDTH) |of| DESTINATION))
(SETQ DESTINATION-BITS-PER-PIXEL (|fetch| (BITMAP BITMAPBITSPERPIXEL) |of| DESTINATION))
(SETQ DESTINATION-BASE (|fetch| (BITMAP BITMAPBASE) |of| DESTINATION))
;; clipping region is initially all of the destination. Clipping coordinates are *inclusive* left and bottom, exclusive right and top -- origin 0.
(SETQ CLIP-LEFT 0)
(SETQ CLIP-BOTTOM 0)
(SETQ CLIP-RIGHT DESTINATION-WIDTH)
(SETQ CLIP-TOP (|fetch| (BITMAP BITMAPHEIGHT) |of| DESTINATION))
;; reduce the region if required by specified destination left, bottom, width, or height
(SETQ CLIP-LEFT (IMAX CLIP-LEFT DESTINATION-LEFT))
(SETQ CLIP-BOTTOM (IMAX CLIP-BOTTOM DESTINATION-BOTTOM))
(COND
  (WIDTH (SETQ CLIP-RIGHT (IMIN (IPLUS DESTINATION-LEFT WIDTH)
                                CLIP-RIGHT))))
(COND
  (HEIGHT (SETQ CLIP-TOP (IMIN (IPLUS DESTINATION-BOTTOM HEIGHT)
                               CLIP-TOP))))
;;
(SETQ S-TO-D-X (IDIFFERENCE DESTINATION-LEFT SOURCE-LEFT))
(SETQ S-TO-D-Y (IDIFFERENCE DESTINATION-BOTTOM SOURCE-BOTTOM))
;; reduce the region if required by source size. We know source origin is (0,0)
(SETQ CLIP-LEFT (IMAX S-TO-D-X CLIP-LEFT)) ; was 0
(SETQ CLIP-BOTTOM (IMAX S-TO-D-Y CLIP-BOTTOM)) ; was 0
(SETQ CLIP-RIGHT (IMIN (IPLUS S-TO-D-X SOURCE-WIDTH)
                       CLIP-RIGHT))
(SETQ CLIP-TOP (IMIN (IPLUS S-TO-D-Y SOURCE-HEIGHT)
                    CLIP-TOP))
;; calculate width and height of clipped region
(SETQ CLIP-WIDTH (IDIFFERENCE CLIP-RIGHT CLIP-LEFT))
(SETQ CLIP-HEIGHT (IDIFFERENCE CLIP-TOP CLIP-BOTTOM))
(COND
  ((OR (ILEQ CLIP-WIDTH 0)
        (ILEQ CLIP-HEIGHT 0))
   ; nothing to do

```

```

(RETURN)))
;; "align" the source file and destination base so that we need only pass in pixel offsets, width, and height
(SETQ DESTINATION-BASE (\\ADDBASE DESTINATION-BASE (ITIMES DESTINATION-RASTER-WIDTH
                                                                (|\\SFInvert| DESTINATION CLIP-TOP))))
(\\SETFILEPTR STREAM (IPLUS (\\GETFILEPTR STREAM)
                             (ITIMES SOURCE-RASTER-WIDTH BYTESPERWORD (- SOURCE-HEIGHT
                                                                           (- CLIP-TOP S-TO-D-Y)))))
(SETQ SOURCE-PIXEL-OFFSET (- CLIP-LEFT S-TO-D-X))
;;
(SELECTQ SOURCE-BITS-PER-PIXEL
  (8 (SELECTQ DESTINATION-BITS-PER-PIXEL
    (8 (AISBLT8TO8.BITMAP STREAM SOURCE-PIXEL-OFFSET SOURCE-RASTER-WIDTH DESTINATION-BASE
                          CLIP-LEFT DESTINATION-RASTER-WIDTH DESTINATION-WIDTH CLIP-WIDTH CLIP-HEIGHT)

      (1 (SELECTQ HOW
        ((FSA :FSA)
         (AISBLT8TO1FSA.BITMAP STREAM SOURCE-PIXEL-OFFSET SOURCE-RASTER-WIDTH
                               DESTINATION-BASE CLIP-LEFT DESTINATION-RASTER-WIDTH
                               DESTINATION-WIDTH CLIP-WIDTH CLIP-HEIGHT PHOTOMETRY-SENSE))
        ((TRUNCATE :TRUNCATE)
         (AISBLTNT01TRUNCATE.BITMAP STREAM SOURCE-PIXEL-OFFSET SOURCE-RASTER-WIDTH
                                     DESTINATION-BASE CLIP-LEFT DESTINATION-RASTER-WIDTH
                                     DESTINATION-WIDTH CLIP-WIDTH CLIP-HEIGHT SOURCE-BITS-PER-PIXEL
                                     PHOTOMETRY-SENSE))
        NIL))
      NIL))
  (4 (SELECTQ DESTINATION-BITS-PER-PIXEL
    (1 (SELECTQ HOW
      ((FSA :FSA)
       (AISBLTNT01FSA.BITMAP STREAM SOURCE-PIXEL-OFFSET SOURCE-RASTER-WIDTH
                           DESTINATION-BASE CLIP-LEFT DESTINATION-RASTER-WIDTH
                           DESTINATION-WIDTH CLIP-WIDTH CLIP-HEIGHT SOURCE-BITS-PER-PIXEL
                           PHOTOMETRY-SENSE))
      ((TRUNCATE :TRUNCATE)
       (AISBLTNT01TRUNCATE.BITMAP STREAM SOURCE-PIXEL-OFFSET SOURCE-RASTER-WIDTH
                                   DESTINATION-BASE CLIP-LEFT DESTINATION-RASTER-WIDTH
                                   DESTINATION-WIDTH CLIP-WIDTH CLIP-HEIGHT SOURCE-BITS-PER-PIXEL
                                   PHOTOMETRY-SENSE))
      NIL))
    NIL))
  (1 (SELECTQ DESTINATION-BITS-PER-PIXEL
    (1 (AISBLT1TO1.BITMAP STREAM SOURCE-PIXEL-OFFSET SOURCE-RASTER-WIDTH DESTINATION-BASE
                          CLIP-LEFT DESTINATION-RASTER-WIDTH DESTINATION-WIDTH CLIP-WIDTH CLIP-HEIGHT
                          PHOTOMETRY-SENSE))
    NIL))
  NIL))))

```

**(AISFILEHEADER**

(LAMBDA (STREAM)

; Edited 21-Sep-88 19:05 by Briggs

;; make sure a file is an ais file and put fileptr at beginning of data.

;; returns a property list format description of the file format

(LET (HEADERLENGTH ATTRIBUTE-PART-HEADER ATTRIBUTE-FILE-POINTER SCRATCH CODING-TYPE)

(\\SETFILEPTR STREAM 0)

(|if| (NEQ (\\WIN STREAM)

AIS-PASSWORD)

|then|

; not an AIS file

NIL

|else| (SETQ HEADERLENGTH (ITIMES (\\WIN STREAM)

; length in bytes

BYTESPERWORD))

(PROG1 (|while| (AND (< (\\GETFILEPTR STREAM)

HEADERLENGTH)

(NOT (EQ 0 (SETQ ATTRIBUTE-PART-HEADER (\\WIN STREAM)))))

|collect| (SETQ ATTRIBUTE-FILE-POINTER (\\GETFILEPTR STREAM))

(PROG1 (SELECTC (LRSH ATTRIBUTE-PART-HEADER 10)

(AIS-RASTER-TYPE ;; The raster part of an AIS file is mandatory

(SETQ SCRATCH (LIST :SCAN-COUNT (\\WIN STREAM)

:SCAN-LENGTH

(\\WIN STREAM)

:SCAN-DIRECTION

(\\WIN STREAM)

:SAMPLES-PER-PIXEL

(\\WIN STREAM)

:CODING-TYPE

(SETQ CODING-TYPE (\\WIN STREAM))))

;; UnCompressedArray is the only known coding type

(SELECTC CODING-TYPE

(AIS-RASTER-CODING-UCA

(LISTPUT SCRATCH :BITS-PER-SAMPLE (\\WIN STREAM))

(LISTPUT SCRATCH :WORDS-PER-SCAN-LINE (\\WIN STREAM))

(LISTPUT SCRATCH :SCAN-LINES-PER-BLOCK

(SIGNED (\\WIN STREAM))

```

                                BITSPERWORD))
                                (LISTPUT SCRATCH :PADDING-PER-BLOCK (SIGNED
                                                                (\\WIN STREAM)
                                                                BITSPERWORD)))
                                NIL)
                                (LIST :RASTER SCRATCH))
(AIS-PLACEMENT-TYPE
  (LIST :PLACEMENT (LIST :LEFT (\\WIN STREAM)
                          :BOTTOM
                          (\\WIN STREAM)
                          :WIDTH
                          (\\WIN STREAM)
                          :HEIGHT
                          (\\WIN STREAM))))
(AIS-PHOTOMETRY-TYPE ;; Ignoring the optional photometry histogram data
  (LIST :PHOTOMETRY (LIST :SIGNAL (\\WIN STREAM)
                           :SENSE
                           (\\WIN STREAM)
                           :SCALE
                           (\\WIN STREAM)
                           :SCALE-A
                           (CONS (SIGNED (\\WIN STREAM)
                                   BITSPERWORD)
                                   (SIGNED (\\WIN STREAM)
                                           BITSPERWORD)))
                           :SCALE-B
                           (CONS (SIGNED (\\WIN STREAM)
                                   BITSPERWORD)
                                   (SIGNED (\\WIN STREAM)
                                           BITSPERWORD)))
                           :SCALE-C
                           (CONS (SIGNED (\\WIN STREAM)
                                   BITSPERWORD)
                                   (SIGNED (\\WIN STREAM)
                                           BITSPERWORD)))
                           :SPOT-TYPE
                           (SIGNED (\\WIN STREAM)
                                   BITSPERWORD)
                           :SPOT-WIDTH
                           (SIGNED (\\WIN STREAM)
                                   BITSPERWORD)
                           :SPOT-LENGTH
                           (SIGNED (\\WIN STREAM)
                                   BITSPERWORD)
                           :SAMPLE-MIN
                           (\\WIN STREAM)
                           :SAMPLE-MAX
                           (\\WIN STREAM))))
(AIS-COMMENT-TYPE ;; (SETQ SCRATCH (ALLOCSTRING (BIN STREAM))) (LIST
;; :COMMENT (AIN SCRATCH 0 (NCHARS SCRATCH)))
                                NIL)
                                NIL)
                                (\\SETFILEPTR STREAM (PLUS ATTRIBUTE-FILE-POINTER
                                                                (ITIMES (SUB1 (LOGAND ATTRIBUTE-PART-HEADER 1023)
                                                                )
                                                                BYTESPERWORD))))
                                (\\SETFILEPTR STREAM HEADERLENGTH))))

```

**WRITEAIS**

(LAMBDA (BITMAP FILE REGION)

; Edited 21-Sep-88 18:34 by Briggs

;; writes a bitmap on to a file in AIS format.

;; simple checks on the arguments before we proceed

```

(OR (TYPEP BITMAP 'BITMAP)
    (\\ILLEGAL.ARG BITMAP))
(OR (AND REGION (REGIONP REGION))
    (AND REGION (\\ILLEGAL.ARG REGION)))
(PROG (STREAM TEMP-BITMAP BITSPERPIXEL RASTERWIDTH WIDTH HEIGHT)
  (SETQ BITSPERPIXEL (|fetch| (BITMAP BITMAPBITSPERPIXEL) |of| BITMAP))
  (COND
    ((REGIONP REGION)

```

;; Get copy of selected REGION of BITMAP into temporary bitmap to avoid having to deal with odd boundary problems when writing  
 ;; contents of BITMAP to STREAM \*

```

    (SETQ TEMP-BITMAP (BITMAPCREATE (|fetch| (REGION WIDTH) |of| REGION)
                                     (|fetch| (REGION HEIGHT) |of| REGION)
                                     BITSPERPIXEL))
    (BITBLT BITMAP (|fetch| (REGION LEFT) |of| REGION)
            (|fetch| (REGION BOTTOM) |of| REGION)
            TEMP-BITMAP)
    (SETQ BITMAP TEMP-BITMAP)))
(SETQ RASTERWIDTH (|fetch| (BITMAP BITMAPRASTERWIDTH) |of| BITMAP))
(SETQ HEIGHT (|fetch| (BITMAP BITMAPHEIGHT) |of| BITMAP))

```

```
(SETQ WIDTH (|fetch| (BITMAP BITMAPWIDTH) |of| BITMAP))
(SETQ STREAM (OPENSTREAM FILE 'OUTPUT))
(\\WOUT STREAM AIS-PASSWORD) ; write AIS password
(\\WOUT STREAM (FOLDLO AIS-DEFAULT-HEADER-LENGTH BYTESPERWORD))
```

:: Generate raster part

```
(\\WOUT STREAM (LOGOR (LLSH AIS-RASTER-TYPE 10)
10)) ; set type and length of raster part header
(\\WOUT STREAM HEIGHT) ; Scan count
(\\WOUT STREAM WIDTH) ; ScanLength
(\\WOUT STREAM 3) ; Scan Dir
(\\WOUT STREAM 1) ; samples per pixel.
(\\WOUT STREAM 1) ; coding type: UnCompressedArray
(\\WOUT STREAM BITSPERPIXEL) ; bits per sample
(\\WOUT STREAM RASTERWIDTH) ; words per sample line.
(\\WOUT STREAM (UNSIGNED -1 16)) ; Sample lines per block: no blocks is 16 bit -1
(\\WOUT STREAM (UNSIGNED -1 16)) ; padding words per block: no blocks is 16 bit -1
```

:: Generate photometry part

```
(\\WOUT STREAM (LOGOR (LLSH AIS-PHOTOMETRY-TYPE 10)
16))
(\\WOUT STREAM AIS-PHOTOMETRY-SIGNAL-BLACK-AND-WHITE)
(\\WOUT STREAM AIS-PHOTOMETRY-SENSE-LARGER-DARKER)
(\\WOUT STREAM AIS-PHOTOMETRY-SCALE-UNSPECIFIED)
(\\WOUT STREAM AIS-PHOTOMETRY-UNSPECIFIED)
(\\WOUT STREAM AIS-PHOTOMETRY-UNSPECIFIED)
(\\WOUT STREAM AIS-PHOTOMETRY-UNSPECIFIED)
(\\WOUT STREAM AIS-PHOTOMETRY-UNSPECIFIED)
(\\WOUT STREAM AIS-PHOTOMETRY-UNSPECIFIED)
(\\WOUT STREAM AIS-PHOTOMETRY-UNSPECIFIED)
(\\WOUT STREAM AIS-PHOTOMETRY-UNSPECIFIED)
(\\WOUT STREAM AIS-PHOTOMETRY-SPOT-TYPE-UNSPECIFIED)
(\\WOUT STREAM AIS-PHOTOMETRY-UNSPECIFIED)
(\\WOUT STREAM AIS-PHOTOMETRY-UNSPECIFIED)
(\\WOUT STREAM 0) ; sample min
(\\WOUT STREAM 1) ; sample max
(\\WOUT STREAM 0) ; no histogram
```

:: position to start of data

```
(\\SETFILEPTR STREAM AIS-DEFAULT-HEADER-LENGTH)
(\\BOUNTS STREAM (|fetch| (BITMAP BITMAPBASE) |of| BITMAP)
0
(ITIMES HEIGHT RASTERWIDTH BYTESPERWORD))
(RETURN (CLOSEF STREAM)))
```

)

:: Internal functions

```
(DEFINEQ
```

**(AISBLT1TO1.BITMAP**

```
(LAMBDA (STREAM SOURCE-PIXEL-OFFSET SOURCE-RASTER-WIDTH DESTINATION-BASE DESTINATION-PIXEL-OFFSET
DESTINATION-RASTER-WIDTH DESTINATION-WIDTH WIDTH HEIGHT PHOTOMETRY-SENSE)
; Edited 22-Sep-88 10:58 by Briggs
```

::: Internal function called by AISBLT.BITMAP to move 1 bpp source file to 1 bpp bitmap

```
(LET ((SOURCE-BYTES-PER-LINE (ITIMES SOURCE-RASTER-WIDTH 2))
(DESTINATION-BYTE-OFFSET)
(WIDTH-BYTES)
(PILOT-BBT (|create| PILOTBBT))
(SCRATCH-BITMAP)
(SCRATCH-BITMAP-BASE))
```

:: look for some special cases that we can handle much faster

```
(COND
((AND (EQ SOURCE-RASTER-WIDTH DESTINATION-RASTER-WIDTH)
(EQ SOURCE-PIXEL-OFFSET 0)
(EQ DESTINATION-PIXEL-OFFSET 0)
(EQ WIDTH DESTINATION-WIDTH))
; source and destination have same raster width
; and the full scan line is being moved to the full destination scan line
(\\BINS STREAM DESTINATION-BASE 0 (ITIMES HEIGHT SOURCE-BYTES-PER-LINE)))
((AND (EQ (IMOD SOURCE-PIXEL-OFFSET BITSPERBYTE)
0)
(EQ (IMOD DESTINATION-PIXEL-OFFSET BITSPERBYTE)
0)
(OR (EQ (IMOD WIDTH BITSPERBYTE)
0)
(EQ WIDTH DESTINATION-WIDTH))))
```

:: Pixel offsets give byte alignment, and the width is an integral number of bytes, or is the destination width (we can run into the slack  
:: bits in the last word with no problem)

```
(SETQ DESTINATION-BYTE-OFFSET (FOLDHI DESTINATION-PIXEL-OFFSET BITSPERBYTE))
(SETQ WIDTH-BYTES (FOLDHI WIDTH BITSPERBYTE))
```

```
(|for| ROW |from| 1 |to| HEIGHT |as| FILE-POINTER |from| (IPLUS (FOLDHI SOURCE-PIXEL-OFFSET BITSPERBYTE)
                                                                (\\GETFILEPTR STREAM))
  |by| SOURCE-BYTES-PER-LINE |bind| (LINE-BASE _ DESTINATION-BASE)
  |do| (\\SETFILEPTR STREAM FILE-POINTER)
      (\\BINS STREAM LINE-BASE DESTINATION-BYTE-OFFSET WIDTH-BYTES)
      (COND
        ((NOT (EQ ROW HEIGHT))
         (SETQ LINE-BASE (\\ADDBASE LINE-BASE DESTINATION-RASTER-WIDTH))))))
(T ;; We have to do bit level realignment -- use a temporary bitmap and let Pilot bitblt deal with it
  (SETQ SCRATCH-BITMAP (BITMAPCREATE WIDTH 1 1))
  (SETQ SCRATCH-BITMAP-BASE (|fetch| (BITMAP BITMAPBASE) |of| SCRATCH-BITMAP))
  (SETUPPILOTBBT PILOT-BBT (|fetch| (BITMAP BITMAPBASE) |of| SCRATCH-BITMAP)
    0
    (UNFOLD (FOLDHI WIDTH BITSPERWORD)
             BITSPERWORD)
    DESTINATION-BASE DESTINATION-PIXEL-OFFSET (UNFOLD DESTINATION-RASTER-WIDTH BITSPERWORD)
    WIDTH 1 0 T T 'INPUT 'REPLACE)
  (SETQ WIDTH-BYTES (FOLDHI WIDTH BITSPERBYTE))
  (|for| ROW |from| 1 |to| HEIGHT |as| FILE-POINTER |from| (IPLUS (FOLDHI SOURCE-PIXEL-OFFSET BITSPERBYTE)
                                                                (\\GETFILEPTR STREAM))
    |by| SOURCE-BYTES-PER-LINE |bind| (LINE-BASE _ DESTINATION-BASE)
    |do| (\\SETFILEPTR STREAM FILE-POINTER)
        (\\BINS STREAM SCRATCH-BITMAP-BASE 0 WIDTH-BYTES)
        (\\PILOTBITBLT PILOT-BBT NIL)
        (COND
          ((NOT (EQ ROW HEIGHT))
           (|freplace| (PILOTBBT PBTDEST) |of| PILOT-BBT |with| (SETQ LINE-BASE (\\ADDBASE LINE-BASE
                                                                                   DESTINATION-RASTER-WIDTH
                                                                                   ))))))))
  (|if| (EQ PHOTOMETRY-SENSE AIS-PHOTOMETRY-SENSE-LARGER-LIGHTER)
    |then| (SETUPPILOTBBT PILOT-BBT DESTINATION-BASE DESTINATION-PIXEL-OFFSET (UNFOLD
                                                                                   DESTINATION-RASTER-WIDTH
                                                                                   BITSPERWORD)
                                                                                   DESTINATION-BASE DESTINATION-PIXEL-OFFSET (UNFOLD DESTINATION-RASTER-WIDTH BITSPERWORD)
                                                                                   WIDTH HEIGHT 0 NIL NIL 'INVERT 'REPLACE)
          (\\PILOTBITBLT PILOT-BBT NIL))))))
```

(AISBLT8TO1FSA.BITMAP

```
(LAMBDA (STREAM SOURCE-PIXEL-OFFSET SOURCE-RASTER-WIDTH DESTINATION-BASE DESTINATION-PIXEL-OFFSET
          DESTINATION-RASTER-WIDTH DESTINATION-WIDTH WIDTH HEIGHT PHOTOMETRY-SENSE)
  ; Edited 23-Sep-88 20:01 by Briggs
  ; Edited 2-May-88 17:00 by Briggs
```

;;; Internal function called by AISBLT.BITMAP to move 8 bpp source file to 1 bpp bitmap using Floyd-Steinberg algorithm

```
;;
;; Use of the Error Table
;;
;; See Newman & Sproull, Principles of Interactive Computer Graphics, pg. 226 for a description of the Floyd-Steinberg algorithm.
;;
;; The error for the current pixel being processed (0<= n < WIDTH) is maintained in ERROR-CURRENT-PIXEL. The error for the pixel directly
;; below the current pixel is stored in ERROR-TABLE[n], while ERROR-TABLE[n+1] represents the error for the pixel to the right. Once
;; ERROR-CURRENT-PIXEL has been used in the calculation it is loaded from ERROR-TABLE[n+1], which frees this cell in the error table to hold
;; the error for the pixel below and to the right of the current pixel.
;;
;;
(LET ((SOURCE-BYTES-PER-LINE (UNFOLD SOURCE-RASTER-WIDTH BYTESPERWORD))
      (INTERMEDIATE-WORD-BUFFER (\\ALLOCBLOCK (FOLDHI WIDTH WORDSPERCELL)))
      (INTERMEDIATE-WORD-BASE)
      (ERROR-CURRENT-PIXEL)
      (ERROR-TABLE (\\ALLOCBLOCK (ADD1 WIDTH)))
      (ERROR-BASE)
      (16-TO-1-PILOTBBT (|create| PILOTBBT))
      (PIXEL)
      (ERROR)
      (QUARTER-ERROR)
      (THREE-EIGHTHS-ERROR))
  ;; Setup for turning words to final destination bits. Note that we conditionally invert the bits -- if the AIS file had bits in the sense 0=black,
  ;; 255=white, because Lisp bitmaps are 0=white, 1=black when displayed.
  (SETUPPILOTBBT 16-TO-1-PILOTBBT INTERMEDIATE-WORD-BUFFER 15 16 DESTINATION-BASE
    DESTINATION-PIXEL-OFFSET 1 1 WIDTH 0 T T (COND
      ((EQ PHOTOMETRY-SENSE
            AIS-PHOTOMETRY-SENSE-LARGER-LIGHTER)
       'INVERT)
      (T 'INPUT))
    'REPLACE)
  ;; clear the error table initially
  (|for| COLUMN |from| 0 |to| (TIMES 2 WIDTH) |by| 2 |do| (\\PUTBASEPTR ERROR-TABLE COLUMN 0))
  ;;
  (|for| ROW |from| 1 |to| HEIGHT |as| FILE-POINTER |from| (IPLUS SOURCE-PIXEL-OFFSET (\\GETFILEPTR STREAM))
```

```

|by| SOURCE-BYTES-PER-LINE |do| ;; position the file at the beginning of the new scan line
(\SETFILEPTR STREAM FILE-POINTER)
;; reset the roving pointer in the error table for this row, and load the current pixel error, clearing the
;; table entry since it will be accumulated into.
(SETQ ERROR-BASE ERROR-TABLE)
(SETQ ERROR-CURRENT-PIXEL (\GETBASEPTR ERROR-BASE 0))
(\PUTBASEPTR ERROR-BASE 0 0)
;; reset the roving pointer to the intermediate result buffer
(SETQ INTERMEDIATE-WORD-BASE INTERMEDIATE-WORD-BUFFER)
(|for| COLUMN |from| 1 |to| WIDTH
  |do| ;; take pixel value as read in plus error accumulated to this pixel -- see note re: error
      ;; calculations above
      (SETQ PIXEL (IPLUS ERROR-CURRENT-PIXEL (\BIN STREAM)))
      ;; threshold
      (COND
        ((IGREATERP PIXEL 127)
         (\PUTBASE INTERMEDIATE-WORD-BASE 0 1)
         (SETQ ERROR (IDIFFERENCE PIXEL 255)))
        (T (\PUTBASE INTERMEDIATE-WORD-BASE 0 0)
         (SETQ ERROR (IDIFFERENCE PIXEL 0))))
      ;; distribute the error (3/8ths to each of pixels to right, and down, 1/4 to pixel diagonally
      ;; down)
      ;; we can use fast logical shifts only if we bias the number to make it positive (we use
      ;; a bias of 32768 here)
      ;; calculate 3/8ths error as half of (error - error/4) -- this way we will incur less error due to
      ;; rounding in the error calculation.
      (SETQ QUARTER-ERROR (IDIFFERENCE (LRSH (IPLUS 32768 ERROR)
                                             2)
                                       (LRSH 32768 2)))
      (SETQ THREE-EIGHTHS-ERROR (IDIFFERENCE
                                  (LRSH (IPLUS 32768 (IDIFFERENCE
                                                         ERROR
                                                         QUARTER-ERROR))
                                       1)
                                  (LRSH 32768 1)))
      ;; pre-load the current pixel error so that the next entry in the error table can be used to
      ;; store the error for the next line down
      (SETQ ERROR-CURRENT-PIXEL (\GETBASEPTR ERROR-BASE 2))
      ;; 3/8ths of the error to the right (the next "current")
      (SETQ ERROR-CURRENT-PIXEL (IPLUS ERROR-CURRENT-PIXEL
                                       THREE-EIGHTHS-ERROR))
      ;; 3/8ths of the error down
      (\PUTBASEPTR ERROR-BASE 0 (IPLUS (\GETBASEPTR ERROR-BASE 0)
                                       THREE-EIGHTHS-ERROR))
      ;; 1/4 of the error down to the right
      (\PUTBASEPTR ERROR-BASE 2 QUARTER-ERROR)
      ;; advance the roving pointer for error table
      (SETQ ERROR-BASE (\ADDBASE ERROR-BASE 2))
      ;; advance pointer to intermediate result scan line buffer
      (SETQ INTERMEDIATE-WORD-BASE (\ADDBASE INTERMEDIATE-WORD-BASE
                                             1)))
      ;; Pack the bits from the intermediate scan line buffer into the destination bitmap at the appropriate
      ;; line and advance the destination scan line pointer.
      (|replace| (PILOTBBT PBTDEST) |of| 16-TO-1-PILOTBBT |with| DESTINATION-BASE
                )
      (\PILOTBITBLT 16-TO-1-PILOTBBT NIL)
      (SETQ DESTINATION-BASE (\ADDBASE DESTINATION-BASE
                                       DESTINATION-RASTER-WIDTH)))
T))

```

**(AISBLT8TO8.BITMAP**

```

(LAMBDA (STREAM SOURCE-PIXEL-OFFSET SOURCE-RASTER-WIDTH DESTINATION-BASE DESTINATION-PIXEL-OFFSET
        DESTINATION-RASTER-WIDTH DESTINATION-WIDTH WIDTH HEIGHT)

```

; Edited 28-Apr-88 11:02 by Briggs

;;; Internal function called by AISBLT.BITMAP to move 8 bpp source file to 8 bpp bitmap

;; look for some special cases that we can handle much faster

```

(LET ((SOURCE-BYTES-PER-LINE (ITIMES SOURCE-RASTER-WIDTH 2)))
  (COND
    ((AND (EQ SOURCE-RASTER-WIDTH DESTINATION-RASTER-WIDTH)
          (EQ SOURCE-PIXEL-OFFSET 0)

```

```

(EQ DESTINATION-PIXEL-OFFSET 0)
(EQ WIDTH DESTINATION-WIDTH)
;; source and destination have same raster width
;; and the full scan line is being moved to the full destination scan line
((\BINS STREAM DESTINATION-BASE 0 (ITIMES HEIGHT SOURCE-BYTES-PER-LINE)))
(T (|for| ROW |from| 1 |to| HEIGHT |as| FILE-POINTER |from| (IPLUS SOURCE-PIXEL-OFFSET (\GETFILEPTR STREAM
)))
  |by| SOURCE-BYTES-PER-LINE |do| ((\SETFILEPTR STREAM FILE-POINTER)
    (\BINS STREAM DESTINATION-BASE DESTINATION-PIXEL-OFFSET WIDTH)
    (COND
      ((NOT (EQ ROW HEIGHT))
        (SETQ DESTINATION-BASE (\ADDBASE DESTINATION-BASE
          DESTINATION-RASTER-WIDTH))))))
  )))

```

**(AISBLTNT01FSA.BITMAP**

```

(LAMBDA (STREAM SOURCE-PIXEL-OFFSET SOURCE-RASTER-WIDTH DESTINATION-BASE DESTINATION-PIXEL-OFFSET
  DESTINATION-RASTER-WIDTH DESTINATION-WIDTH WIDTH HEIGHT SOURCE-BITS-PER-PIXEL PHOTOMETRY-SENSE)
  ; Edited 23-Sep-88 20:11 by Briggs
  ; Edited 2-May-88 15:40 by Briggs

```

;;; Internal function called by AISBLT.BITMAP to move N bpp source file to 1 bpp bitmap using Floyd-Steinberg algorithm. For N=8, use the special case  
;;; version AISBLT8TO1FSA.BITMAP.

```

;;
;; Use of the Error Table
;;
;; See Newman & Sproull, Principles of Interactive Computer Graphics, pg. 226 for a description of the Floyd-Steinberg algorithm.
;;
;; The error for the current pixel being processed (0<= n < WIDTH) is maintained in ERROR-CURRENT-PIXEL. The error for the pixel directly
;; below the current pixel is stored in ERROR-TABLE[n], while ERROR-TABLE[n+1] represents the error for the pixel to the right. Once
;; ERROR-CURRENT-PIXEL has been used in the calculation it is loaded from ERROR-TABLE[n+1], which frees this cell in the error table to hold
;; the error for the pixel below and to the right of the current pixel.
;;

```

```

(LET* ((SOURCE-BYTES-PER-LINE (ITIMES SOURCE-RASTER-WIDTH 2))
  (SOURCE-LINE-BYTE-BUFFER-BASE (\ALLOCBLOCK (FOLDHI SOURCE-RASTER-WIDTH WORDSPERCELL)))
  (INTERMEDIATE-WORD-BUFFER (\ALLOCBLOCK (FOLDHI WIDTH WORDSPERCELL)))
  (INTERMEDIATE-WORD-BASE)
  (ERROR-CURRENT-PIXEL)
  (ERROR-TABLE (\ALLOCBLOCK (ADD1 WIDTH)))
  (ERROR-BASE)
  (ERROR-FRACTIONAL-POINT 7)
  (N-TO-16-PILOTBBT (|create| PILOTBBT))
  (16-TO-1-PILOTBBT (|create| PILOTBBT))
  (PIXEL)
  (BLACK 0)
  (WHITE (SUB1 (EXPT 2 SOURCE-BITS-PER-PIXEL)))
  (THRESHOLD (LRSH (IPLUS BLACK WHITE)
    1))
  (ERROR)
  (QUARTER-ERROR)
  (THREE-EIGHTHS-ERROR))

```

;; do the setup for expanding source pixels to words.

```

(SETUPPILOTBBT N-TO-16-PILOTBBT SOURCE-LINE-BYTE-BUFFER-BASE (ITIMES SOURCE-PIXEL-OFFSET
  SOURCE-BITS-PER-PIXEL)
  SOURCE-BITS-PER-PIXEL INTERMEDIATE-WORD-BUFFER (IDIFFERENCE 16 SOURCE-BITS-PER-PIXEL)
  16 SOURCE-BITS-PER-PIXEL WIDTH 0 T T 'INPUT 'REPLACE)

```

;; Setup for turning words to final destination bits. Note that we conditionally invert the bits -- if the AIS file had bits in the sense 0=black,  
;; larger=lighter, because Lisp bitmaps are 0=white, 1=black when displayed.

```

(SETUPPILOTBBT 16-TO-1-PILOTBBT INTERMEDIATE-WORD-BUFFER 15 16 DESTINATION-BASE
  DESTINATION-PIXEL-OFFSET 1 1 WIDTH 0 T T (COND
    ((EQ PHOTOMETRY-SENSE
      AIS-PHOTOMETRY-SENSE-LARGER-LIGHTER)
      'INVERT)
    (T 'INPUT))
  'REPLACE)

```

;; clear the error table initially

```

(|for| COLUMN |from| 0 |to| (TIMES 2 WIDTH) |by| 2 |do| (\PUTBASEPTR ERROR-TABLE COLUMN 0))
;;

```

(|for| ROW |from| 1 |to| HEIGHT |do| ;; We read a full scan line, and extract the bits we need as we expand to 16 bits per pixel

```

  (\BINS STREAM SOURCE-LINE-BYTE-BUFFER-BASE 0 SOURCE-BYTES-PER-LINE)
  ;; expand the pixels to words to make them easier to deal with
  (\PILOTBITBLT N-TO-16-PILOTBBT NIL)

```

;; reset the roving pointer in the error table for this row, and load the current pixel error, resetting the  
;; table entry to 0 because it will be accumulated to as error for the next line



```

(SETQ ERROR-BASE ERROR-TABLE)
(SETQ ERROR-CURRENT-PIXEL (\\GETBASEPTR ERROR-BASE 0))
(\\PUTBASEPTR ERROR-BASE 0 0)
;; reset the roving pointer to the intermediate result buffer
(SETQ INTERMEDIATE-WORD-BASE INTERMEDIATE-WORD-BUFFER)
(|for| COLUMN |from| 1 |to| WIDTH
  |do| ;; take pixel value as read in plus error accumulated to this pixel
    (SETQ PIXEL (IPLUS ERROR-CURRENT-PIXEL (\\GETBASE
                                             INTERMEDIATE-WORD-BASE
                                             0)))
    ;; threshold
    (COND
      ((IGREATERP PIXEL THRESHOLD)
       (\\PUTBASE INTERMEDIATE-WORD-BASE 0 1)
       (SETQ ERROR (IDIFFERENCE PIXEL WHITE)))
      (T (\\PUTBASE INTERMEDIATE-WORD-BASE 0 0)
         (SETQ ERROR (IDIFFERENCE PIXEL BLACK))))
    ;; distribute the error (3/8ths to each of pixels to right, and down, 1/4 to pixel diagonally
    ;; down)
    ;; we can use fast logical shifts only if we bias the number to make it positive
    (SETQ QUARTER-ERROR (IDIFFERENCE (LRSH (IPLUS 32768 ERROR)
                                           2)
                                     (LRSH 32768 2)))
    (SETQ THREE-EIGHTHS-ERROR (IDIFFERENCE
                                (LRSH (IPLUS 32768 (IDIFFERENCE ERROR
                                                         QUARTER-ERROR)
                                         1)
                                      (LRSH 32768 1))))
    ;; pre-load the current pixel error so that the next entry in the error table can be used to
    ;; store the error for the next line down
    (SETQ ERROR-CURRENT-PIXEL (\\GETBASEPTR ERROR-BASE 2))
    ;; 3/8ths of the error to the right (the next "current")
    (SETQ ERROR-CURRENT-PIXEL (IPLUS ERROR-CURRENT-PIXEL
                                     THREE-EIGHTHS-ERROR))
    ;; 3/8ths of the error down
    (\\PUTBASEPTR ERROR-BASE 0 (IPLUS (\\GETBASEPTR ERROR-BASE 0)
                                     THREE-EIGHTHS-ERROR))
    ;; 1/4 of the error down to the right
    (\\PUTBASEPTR ERROR-BASE 2 QUARTER-ERROR)
    ;; advance the roving pointer for error table
    (SETQ ERROR-BASE (\\ADDBASE ERROR-BASE 2))
    ;; advance pointer to intermediate result scan line buffer
    (SETQ INTERMEDIATE-WORD-BASE (\\ADDBASE INTERMEDIATE-WORD-BASE 1
                                           )))
  ;; Pack the bits from the intermediate scan line buffer into the destination bitmap at the appropriate
  ;; line and advance the destination scan line pointer.
  (|freplace| (PILOTBBT PBTDEST) |of| 16-TO-1-PILOTBBT |with| DESTINATION-BASE)
  (\\PILOTBITBLT 16-TO-1-PILOTBBT NIL)
  (SETQ DESTINATION-BASE (\\ADDBASE DESTINATION-BASE
                                  DESTINATION-RASTER-WIDTH)))
T))

```

**(AISBLTNT01TRUNCATE.BITMAP**

```

(LAMBDA (STREAM SOURCE-PIXEL-OFFSET SOURCE-RASTER-WIDTH DESTINATION-BASE DESTINATION-PIXEL-OFFSET
        DESTINATION-RASTER-WIDTH DESTINATION-WIDTH WIDTH HEIGHT SOURCE-BITS-PER-PIXEL PHOTOMETRY-SENSE)
  ; Edited 22-Sep-88 10:23 by Briggs
  ; Edited 2-May-88 15:40 by Briggs

```

;;; Internal function called by AISBLT.BITMAP to move N bpp source file to 1 bpp bitmap using truncation.

```

(LET* ((SOURCE-BYTES-PER-LINE (ITIMES SOURCE-RASTER-WIDTH 2))
       (SOURCE-LINE-BYTE-BUFFER-BASE (\\ALLOCBLOCK (FOLDHI SOURCE-RASTER-WIDTH WORDSPERCELL)))
       (HIGH-N-TO-1-PILOTBBT (|create| PILOTBBT))
       (16-TO-1-PILOTBBT (|create| PILOTBBT)))
  ;; Setup for turning source pixels to destination pixels. Note that we conditionally invert the bits -- if the AIS file had bits in the sense
  ;; 0=black, larger=lighter, because Lisp bitmaps are 0=white, 1=black when displayed.
  (SETUPPILOTBBT HIGH-N-TO-1-PILOTBBT SOURCE-LINE-BYTE-BUFFER-BASE (ITIMES SOURCE-PIXEL-OFFSET
                                                                    SOURCE-BITS-PER-PIXEL)
                 SOURCE-BITS-PER-PIXEL DESTINATION-BASE DESTINATION-PIXEL-OFFSET 1 1 WIDTH 0 T T
  (COND
    ((EQ PHOTOMETRY-SENSE AIS-PHOTOMETRY-SENSE-LARGER-LIGHTER)
     'INVERT)

```

```

      (T 'INPUT))
      'REPLACE)
    (|for| ROW |from| 1 |to| HEIGHT |do| ;; We read a full scan line, and extract the bits we need.
      (\\BINS STREAM SOURCE-LINE-BYTE-BUFFER-BASE 0 SOURCE-BYTES-PER-LINE)
      ;; Pack the bits from the source scan line buffer into the destination bitmap at the appropriate line and
      ;; advance the destination scan line pointer.
      (|freplace| (PILOTBBT PBTDEST) |of| HIGH-N-TO-1-PILOTBBT |with|
        DESTINATION-BASE
      )
      (\\PILOTBITBLT HIGH-N-TO-1-PILOTBBT NIL)
      (SETQ DESTINATION-BASE (\\ADDBASE DESTINATION-BASE
        DESTINATION-RASTER-WIDTH)))
    T))

```

**(SETUPPILOTBBT**

```

(LAMBDA (PILOT-BBT SOURCE-BASE SOURCE-BIT SOURCE-BPL DESTINATION-BASE DESTINATION-BIT DESTINATION-BPL WIDTH
  HEIGHT FLAGS DISJOINT DISJOINT-ITEMS SOURCE OPERATION)
  ; Edited 28-Apr-88 18:21 by Briggs
  (|freplace| (PILOTBBT PBTDEST) |of| PILOT-BBT |with| DESTINATION-BASE)
  (|freplace| (PILOTBBT PBTDESTBIT) |of| PILOT-BBT |with| DESTINATION-BIT)
  (|freplace| (PILOTBBT PBTDESTBPL) |of| PILOT-BBT |with| DESTINATION-BPL)
  (|freplace| (PILOTBBT PBTSOURCE) |of| PILOT-BBT |with| SOURCE-BASE)
  (|freplace| (PILOTBBT PBTSOURCEBIT) |of| PILOT-BBT |with| SOURCE-BIT)
  (|freplace| (PILOTBBT PBTSOURCEBPL) |of| PILOT-BBT |with| SOURCE-BPL)
  (|freplace| (PILOTBBT PBTWIDTH) |of| PILOT-BBT |with| WIDTH)
  (|freplace| (PILOTBBT PBTHEIGHT) |of| PILOT-BBT |with| HEIGHT)
  (|freplace| (PILOTBBT PBTFLAGS) |of| PILOT-BBT |with| FLAGS)
  (|freplace| (PILOTBBT PBTDISJOINT) |of| PILOT-BBT |with| DISJOINT)
  (|freplace| (PILOTBBT PBTDISJOINTITEMS) |of| PILOT-BBT |with| DISJOINT-ITEMS)
  (|freplace| (PILOTBBT PBTUSEGRAY) |of| PILOT-BBT |with| NIL)
  (\\SETPBTFUNCTION PILOT-BBT SOURCE OPERATION)))
)

```

;; Useful constants  
;; AIS file identifying word

```
(CL:DEFCONSTANT AIS-PASSWORD 33962)
```

;; Attribute part header types

```
(CL:DEFCONSTANT AIS-RASTER-TYPE 1)
```

```
(CL:DEFCONSTANT AIS-PLACEMENT-TYPE 2)
```

```
(CL:DEFCONSTANT AIS-PHOTOMETRY-TYPE 3)
```

```
(CL:DEFCONSTANT AIS-COMMENT-TYPE 4)
```

;; Raster coding types

```
(CL:DEFCONSTANT AIS-RASTER-CODING-UCA 1)
```

```
(CL:DEFCONSTANT AIS-RASTER-CODING-CA 2)
```

;; Photometry sense

```
(CL:DEFCONSTANT AIS-PHOTOMETRY-SENSE-LARGER-DARKER 1)
```

```
(CL:DEFCONSTANT AIS-PHOTOMETRY-SENSE-LARGER-LIGHTER 0)
```

;; Photometry signal

```
(CL:DEFCONSTANT AIS-PHOTOMETRY-SIGNAL-BLACK-AND-WHITE 0
  "Photometry signal is black and white")
```

```
(CL:DEFCONSTANT AIS-PHOTOMETRY-SIGNAL-RED-SEPARATION 1
  "Photometry signal is red separation")
```

(CL:DEFCONSTANT **AIS-PHOTOMETRY-SIGNAL-BLUE-SEPARATION** 2  
"Photometry signal is blue separation")

(CL:DEFCONSTANT **AIS-PHOTOMETRY-SIGNAL-GREEN-SEPARATION** 3  
"Photometry signal is green separation")

(CL:DEFCONSTANT **AIS-PHOTOMETRY-SIGNAL-CYAN-SEPARATION** 4  
"Photometry signal is cyan separation")

(CL:DEFCONSTANT **AIS-PHOTOMETRY-SIGNAL-MAGENTA-SEPARATION** 5  
"Photometry signal is magenta separation")

(CL:DEFCONSTANT **AIS-PHOTOMETRY-SIGNAL-YELLOW-SEPARATION** 6  
"Photometry signal is yellow separation")

(CL:DEFCONSTANT **AIS-PHOTOMETRY-SIGNAL-X-CIE** 7  
"Photometry signal is x signal (CIE)")

(CL:DEFCONSTANT **AIS-PHOTOMETRY-SIGNAL-Y-CIE** 8  
"Photometry signal is y signal (CIE)")

(CL:DEFCONSTANT **AIS-PHOTOMETRY-SIGNAL-IN-COMMENT** (UNSIGNED -2 16)  
"Photometry signal specified in comment part")

(CL:DEFCONSTANT **AIS-PHOTOMETRY-SIGNAL-UNSPECIFIED** (UNSIGNED -1 16)  
"Photometry signal unspecified")

:: Photometry spot type

(CL:DEFCONSTANT **AIS-PHOTOMETRY-SPOT-TYPE-RECTANGULAR** 1  
"Photometry spot type is rectangular")

(CL:DEFCONSTANT **AIS-PHOTOMETRY-SPOT-TYPE-CIRCULAR** 2  
"Photometry spot type is circular")

(CL:DEFCONSTANT **AIS-PHOTOMETRY-SPOT-TYPE-IN-COMMENTS** (UNSIGNED -2 16)  
"Photometry spot type is specified in comments")

(CL:DEFCONSTANT **AIS-PHOTOMETRY-SPOT-TYPE-UNSPECIFIED** (UNSIGNED -1 16)  
"Photometry spot type is unspecified")

:: Photometry scale

(CL:DEFCONSTANT **AIS-PHOTOMETRY-SCALE-REFLECTANCE-TRANSMITTANCE** 1  
"Photometry scale is reflectance or transmittance x 1000")

(CL:DEFCONSTANT **AIS-PHOTOMETRY-SCALE-OPTICAL-DENSITY** 2  
"Photometry scale is optical density x 1000")

(CL:DEFCONSTANT **AIS-PHOTOMETRY-SCALE-IN-COMMENT** (UNSIGNED -2 16)  
"Photometry scale is specified in comments")

(CL:DEFCONSTANT **AIS-PHOTOMETRY-SCALE-UNSPECIFIED** (UNSIGNED -1 16)  
"Photometry scale is unspecified")

:: Photometry

(CL:DEFCONSTANT **AIS-PHOTOMETRY-UNSPECIFIED** (UNSIGNED -1 16)  
"Photometry general unspecified value")

:: Header information when writing AIS format

(CL:DEFCONSTANT **AIS-DEFAULT-HEADER-LENGTH** (CL:\* 1024 BYTESPERWORD)  
"Length in bytes of the header to write in AIS files")

(PUTPROPS **AISBLT COPYRIGHT** ("Xerox Corporation" 1988))

---

**FUNCTION INDEX**

AISBLT.BITMAP .....	1	AISBLT8TO8.BITMAP .....	7	AISFILEHEADER .....	3
AISBLT1TO1.BITMAP .....	5	AISBLTNT01FSA.BITMAP .....	8	SETUPPILOTBBT .....	10
AISBLT8TO1FSA.BITMAP .....	6	AISBLTNT01TRUNCATE.BITMAP .....	9	WRITEAIS .....	4

---

**CONSTANT INDEX**

AIS-COMMENT-TYPE .....	10	AIS-PHOTOMETRY-SIGNAL-RED-SEPARATION .....	10
AIS-DEFAULT-HEADER-LENGTH .....	11	AIS-PHOTOMETRY-SIGNAL-UNSPECIFIED .....	11
AIS-PASSWORD .....	10	AIS-PHOTOMETRY-SIGNAL-X-CIE .....	11
AIS-PHOTOMETRY-SCALE-IN-COMMENT .....	11	AIS-PHOTOMETRY-SIGNAL-Y-CIE .....	11
AIS-PHOTOMETRY-SCALE-OPTICAL-DENSITY .....	11	AIS-PHOTOMETRY-SIGNAL-YELLOW-SEPARATION .....	11
AIS-PHOTOMETRY-SCALE-RELECTANCE-TRANSMITTANCE .....	11	AIS-PHOTOMETRY-SPOT-TYPE-CIRCULAR .....	11
AIS-PHOTOMETRY-SCALE-UNSPECIFIED .....	11	AIS-PHOTOMETRY-SPOT-TYPE-IN-COMMENTS .....	11
AIS-PHOTOMETRY-SENSE-LARGER-DARKER .....	10	AIS-PHOTOMETRY-SPOT-TYPE-RECTANGULAR .....	11
AIS-PHOTOMETRY-SENSE-LARGER-LIGHTER .....	10	AIS-PHOTOMETRY-SPOT-TYPE-UNSPECIFIED .....	11
AIS-PHOTOMETRY-SIGNAL-BLACK-AND-WHITE .....	10	AIS-PHOTOMETRY-TYPE .....	10
AIS-PHOTOMETRY-SIGNAL-BLUE-SEPARATION .....	11	AIS-PHOTOMETRY-UNSPECIFIED .....	11
AIS-PHOTOMETRY-SIGNAL-CYAN-SEPARATION .....	11	AIS-PLACEMENT-TYPE .....	10
AIS-PHOTOMETRY-SIGNAL-GREEN-SEPARATION .....	11	AIS-RASTER-CODING-CA .....	10
AIS-PHOTOMETRY-SIGNAL-IN-COMMENT .....	11	AIS-RASTER-CODING-UCA .....	10
AIS-PHOTOMETRY-SIGNAL-MAGENTA-SEPARATION .....	11	AIS-RASTER-TYPE .....	10

---