PS 390 RASTER PROGRAMMING

DISPLAYING HOST-GENERATED IMAGES WITH THE PS 390 RASTER SYSTEM

Supported Under Graphics Firmware A2.V02 and Higher

EVANS & SUTHERLAND

April 1987
E&S #901194-088 P1

CONTENTS

FIGURES AND TABLES

Introduction

The PS 390 raster system consists of a printed circuit card that outputs static images to a 1024 (column) by 864 (row) pixel raster display. Each pixel is 24 bits deep for addressing into a red-green-blue color lookup table that is 24 bits deep.

The PS 390 raster system can be used to display polygon wireframe models and shaded images derived locally from PS 300 polygonal models, or it can be used as a frame buffer to display host-generated images. When used as a frame buffer, the PS 390 only serves as a communications link between the host and the raster system. No standard PS 300 commands or data structures are used to display host-generated images.

This document describes how to display host-generated images using the FORTRAN and Pascal Graphics Support Routines (GSRs) and user-generated routines. Programming examples for both methods are provided.

This manual assumes that you are familiar with creating raster images. If you need background in this subject, the following books contain detailed sections on raster graphics:

J.D. Foley and A. Van Dam: *Fundamentals of Interactive Computer Graphics*. Addison-Wesley Publishing Company, 1982.

William M. Newman and Robert F. Sproul: *Principles of Interactive Computer Graphics*. McGraw-Hill Book Company, 1979.

Conrac Division: *Raster Graphics Handbook*. Conrac Corporation (600 North Rimsdale Avenue, Covina CA, 91722), 1980.

Donald P. Greenburg: *Introduction to Raster Graphics*. Siggraph '83 Tutorial, 1983.

PS 390 Raster Concepts

The basic steps required to display a host-generated picture on the PS 390 are:

- Determine what your picture will look like (determine pixel values and the addresses into the lookup tables).
- Set the logical device coordinates to specify the proper size and position of the raster image.
- Transfer this information from the host to the PS 390 via the Graphics Support Routines or user-written routines.

Three features of the PS 390 image buffer mode are:

- It is run-length encoded.
- It uses red, green, and blue color lookup tables.
- It specifies logical device coordinates to define the portion of virtual address space (the total coordinate area in which pictures can be created) that contains the raster picture. This allows flexibility in positioning a picture relative to the actual screen display.

These concepts and their application in the PS 390 raster system are discussed in detail in the following sections.

Run-Length Encoding

Some raster systems require that you encode a raster picture pixel by pixel. That is, each pixel on the raster screen must be addressed individually. In contrast, the PS 390 accepts raster data from the host in run-length encoded format. Both the Graphics Support Routines and user-written routines specify run-length encoding.

In run-length encoding, a set of consecutive pixels of the same color is specified in a single command containing the number of consecutive pixels and the color value of the pixels. Since, in practice, most pictures contain many sequences of consecutive pixels of the same color, run-length encoding allows more efficient picture transmission than pixel-by-pixel encoding in all but the most complex and high-resolution raster pictures.

For example, if the bottom third of your raster picture is a background color, one run-length encoded command could specify the color for those 294,912 (1024x288) pixels. Pixel-by-pixel encoding would require 294,912 separate single-pixel commands.

Color Lookup Tables

Any displayable color is a combination of three components---red, green, and blue, the primary phosphor colors used in the raster display's additive color process. Varying the intensity of these three color components produces the wide variety of colors available to the raster display.

The PS 390 raster system does not specify colors directly, but rather refers to locations in color lookup tables (LUTs) that contain the color entries. Each pixel on the raster display is 24 bits deep. That is, 24 bits of data address each pixel's color value in the color lookup tables with 8 bits to specify entries in the color lookup tables for each red, green, and blue color. Since the 24 bits of pixel data do not specify a color directly, this is sometimes referred to as "pseudocolor" specification.

There are 3 color lookup tables on the raster card, one each of red, green, and blue. The color lookup table entries (derived from the 24 bits of pixel data) contain a precise color level, or intensity for a specific color.

Each entry in the 3 lookup tables is 8 bits deep, providing 2^{24} potential colors. Each set of 8 bits specifies the intensity of the corresponding red, green, or blue color. Each lookup table has 256 (0–255) possible entries (i.e., 8 bits of address per lookup table), providing 2^{24} (256^3) usable colors. This provides more displayable colors than there are actual pixels on the raster screen. Naturally, the human eye cannot distinguish between this many shades of colors. This permits "smooth shading" of host-generated raster pictures. If the eye could actually perceive the slight differences in shades of colors, you would see banding (stripes of different shades) instead of smooth shading.

These color tables are preloaded at boot time with a gamma-corrected lookup table. This gives the appearance on the screen of a linear change in integrity as the index changes (i.e., location 20 is twice as bright as 10).

Figure 1 provides a graphic representation of how the 24 bits of pixel data map to the 24 bits of the color lookup tables. The top of the figure shows the image buffer memory of the system. Each pixel contains 24 bits of pixel data made up of 8 bits of red, 8 bits of green, and 8 bits of blue pixel data. These bits specify the address in the color lookup tables.

The 8 bit entries in the color lookup tables specify intensities of red, green, and blue (RGB). The 8-bit digital-to-analog converters change these digital values to analog signals which drive the red, green, and blue guns that stimulate the RGB triads on the raster display screen. The eye blends these intensities to generate the specified color.

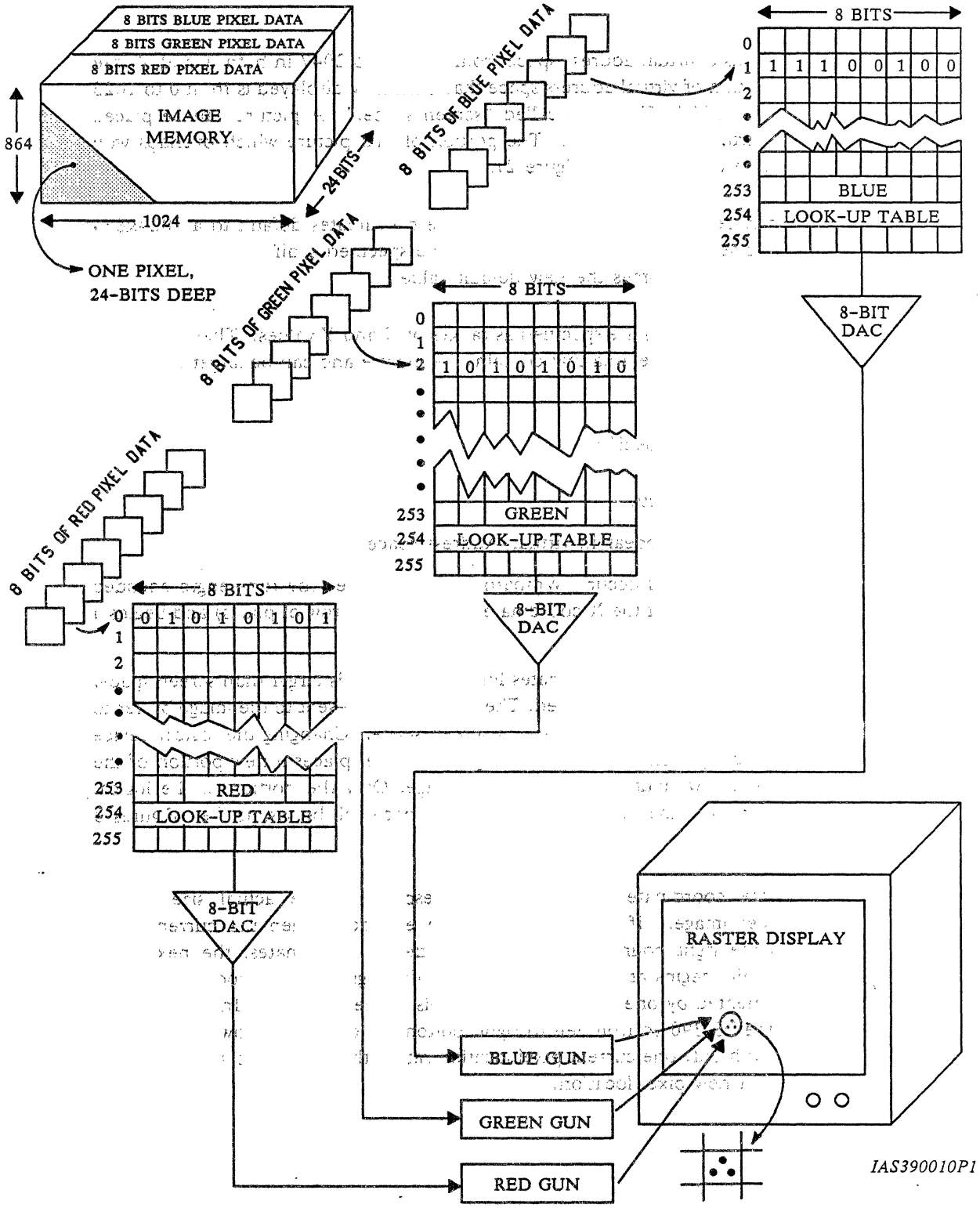


Figure 1. Pixel Mapping to Color Lookup Tables

Logical Device Coordinates

The raster option has a virtual address space from -32768 to 2047 in both X and Y (see Figure 2). The portion of virtual address space that is actually displayed is from 0 to 1023 in X and from 0 to 863 in Y, and is called "screen space." A picture can be placed anywhere in the virtual address space. The portion of that picture which overlaps with screen space will be displayed (see Figure 2).

When the raster system is booted, the logical device coordinates default to a 1024x864 screen size starting at 0,0. Of course, once you have specified a different set of logical device coordinates, this becomes the new default value.

The logical device coordinates are specified as ranges of X and Y values. They define the dimensions and position of the area that contains the picture and can be larger or smaller than screen space.

Logical device coordinates specify:

1. The size of the raster picture.
2. Where the picture will appear in virtual-Address space.
3. Where "wraparound" will occur. Wraparound occurs when the run-length encoded command hits the limit of the X coordinates (the end of a row of pixels) and begins a new row of pixels.

When sending the logical device coordinates for a picture that is larger than screen space, data outside of screen space are discarded. The data can be re-sent to the image buffer to display another portion of the raster image in screen space. Changing the logical device coordinates and starting position, and resending the picture, places a new portion of the image in screen space without recalculating the image. Only the portion of the logical device coordinate picture that coincides with screen space will be visible (see Figures 3 and 4).

The logical device coordinate range should correspond to the actual size of the precalculated raster image. If correctly run-length encoded, when the current pixel location reaches the right boundary of the logical device coordinates, the next pixel location automatically begins at the left boundary of the logical device coordinates with the Y value incremented by one, addressing the pixels in the next row. In other words, raster images in the PS 390 go from left to right, bottom to top. This allows you to send an entire picture with only one current pixel location rather than having to start each new row of pixels with a new pixel location.

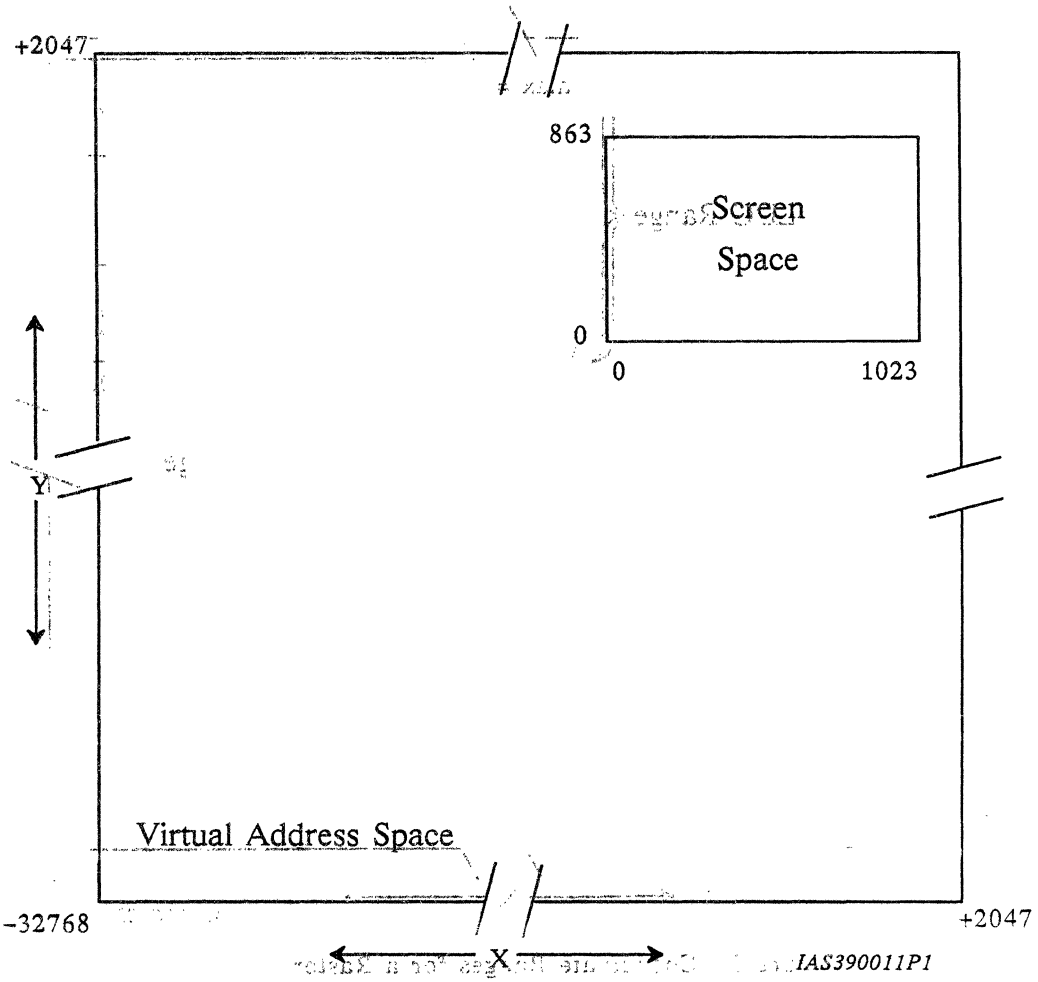


Figure 2. Virtual Address Space and Screen Space

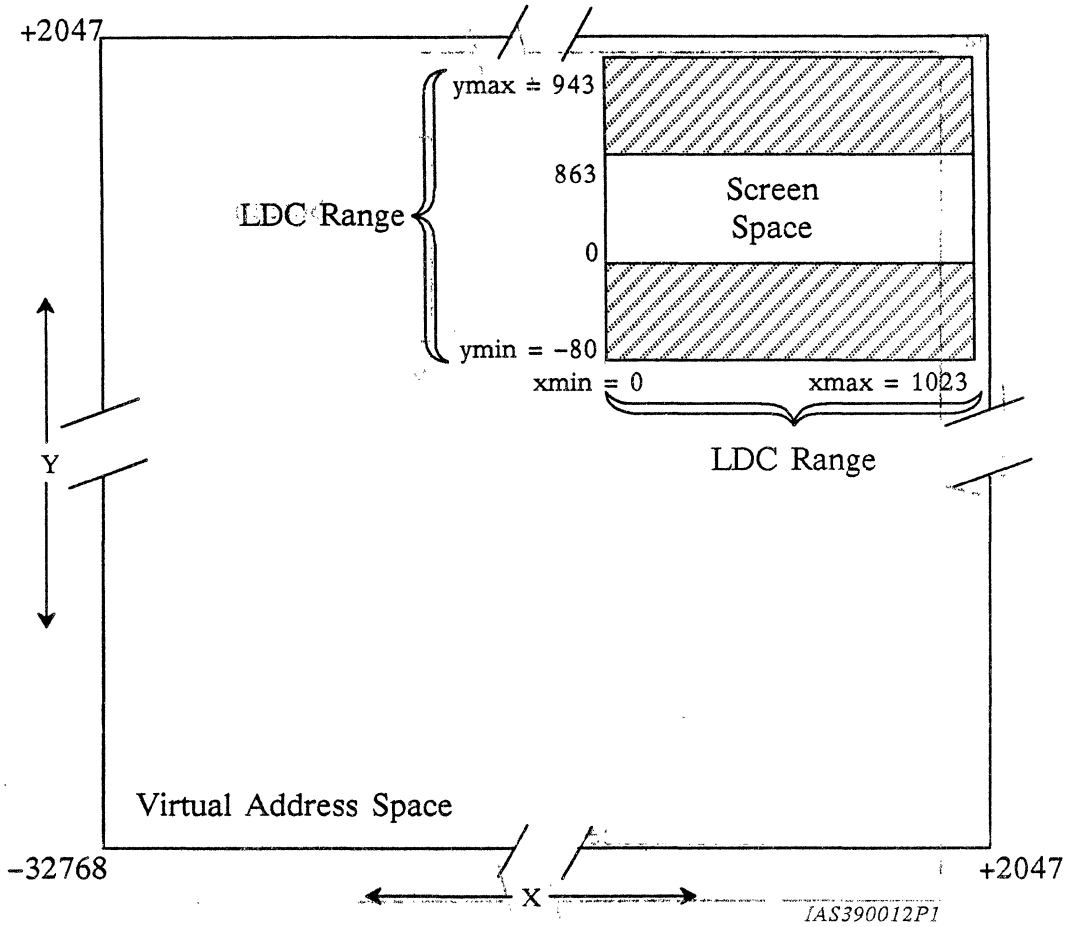


Figure 3. Coordinate Ranges for a Raster Display

Figure 3 illustrates the coordinate ranges for virtual address space (the total coordinate area in which pictures can be created), sample logical device coordinates specified by the programmer (in this case, shown in the shaded area specifying a 1024 x 1024 image), and the actual screen space that can be displayed at any given time.

Run-length encoded commands make no mention of absolute pixel location. The commands simply specify the next (n) consecutive pixels starting at the current pixel location. (The current pixel location is the point in the logical device coordinates where the run-length encoded command begins loading pixels.) It follows that an entire picture can be repositioned by changing the logical device coordinate specifications (which are the only specifications that refer to absolute pixel locations) and retransmitting the picture data that fall in the new logical device coordinates. No change to the encoded pixel data is necessary.

Figure 4 shows virtual address space (the entire area in which a picture can be created), the logical device coordinate range (specified by the WRPIX command described in the

next section), and the screen space containing the portion of the picture that will actually be displayed on the raster monitor.

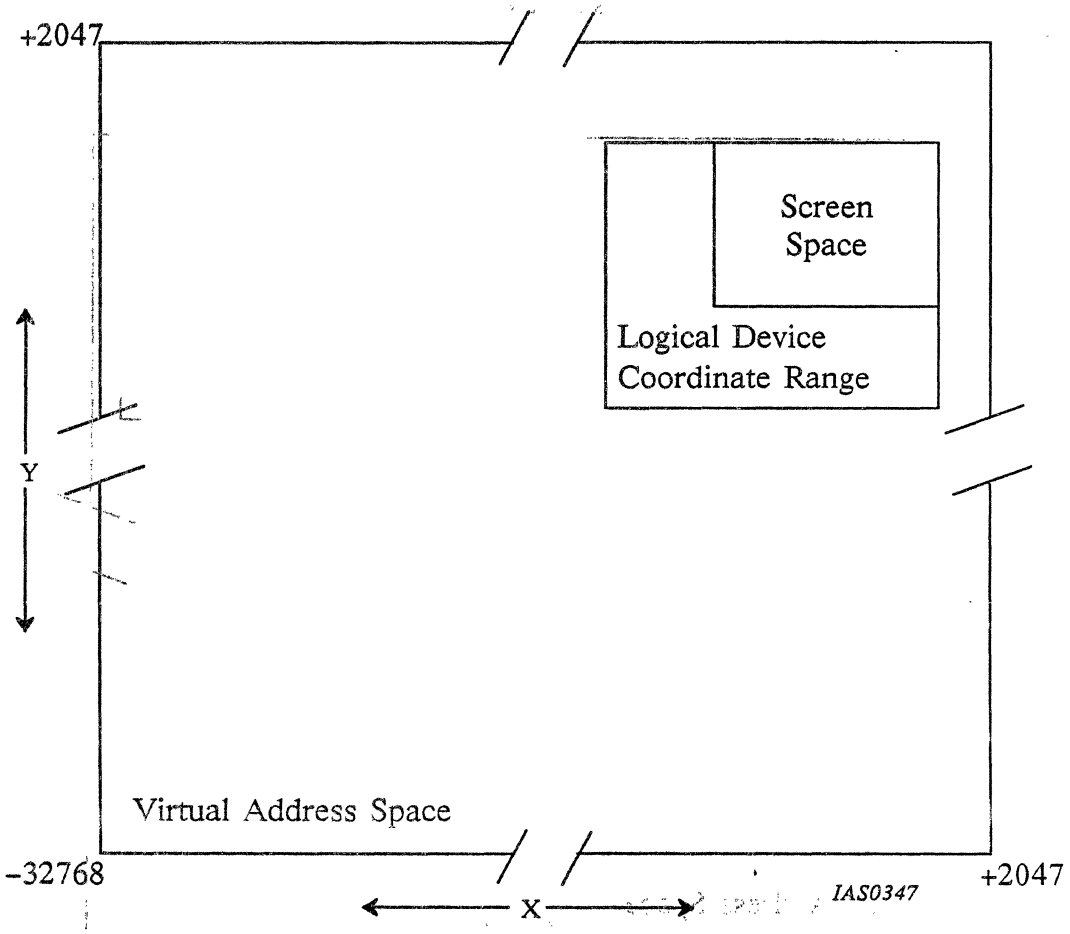


Figure 4. Virtual Address Space, Logical Device Coordinate Range, and Screen Space

Figure 5 shows that the logical device coordinate range has been changed so that the lower left-hand area of the logical device coordinate range coincides with screen space. Note that a new section of the raster image is in screen space after the picture data have been retransmitted. Also, screen space remains fixed: the new logical device coordinate range has changed what actually appears in screen space.

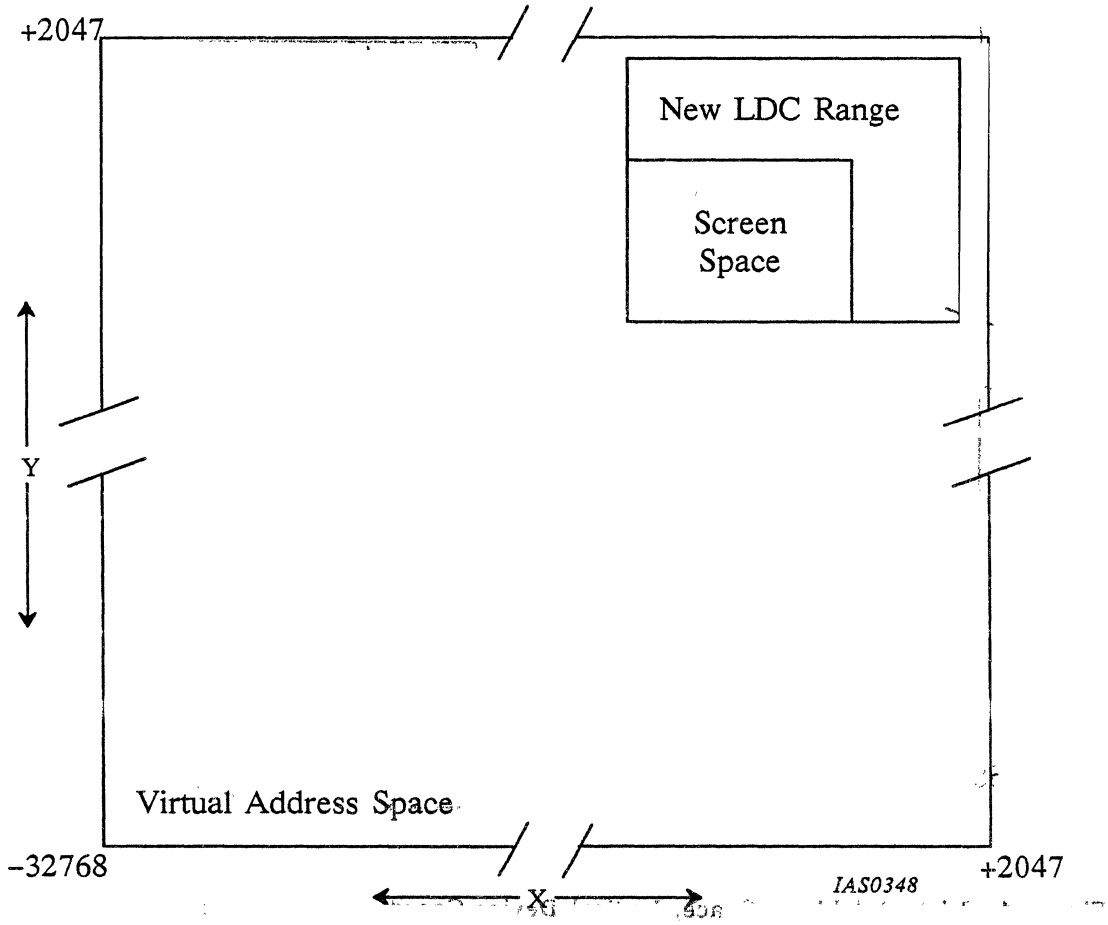


Figure 5. Displayed Raster Image Within Lower Left LDC Range

The raster option has the ability of virtual pixel addressing of:

-32768 <= X <= 2047, -32768 <= Y <= 2047

of which the portion that is actually displayed is:

0 <= X <= 1023, 0 <= Y <= 863

The logical device coordinates (Xmin <= X <= Xmax, Ymin <= Y <= Ymax) can be any subset of this range.

To position a raster image of 200x200 on the center of the screen, the values should be:

Xmin = 412 ([1024-200]/2)
Xmax = 611 (Xmin + 200 - 1)
Ymin = 332 ([864-200]/2)
Ymax = 531 (Ymin + 200 - 1)

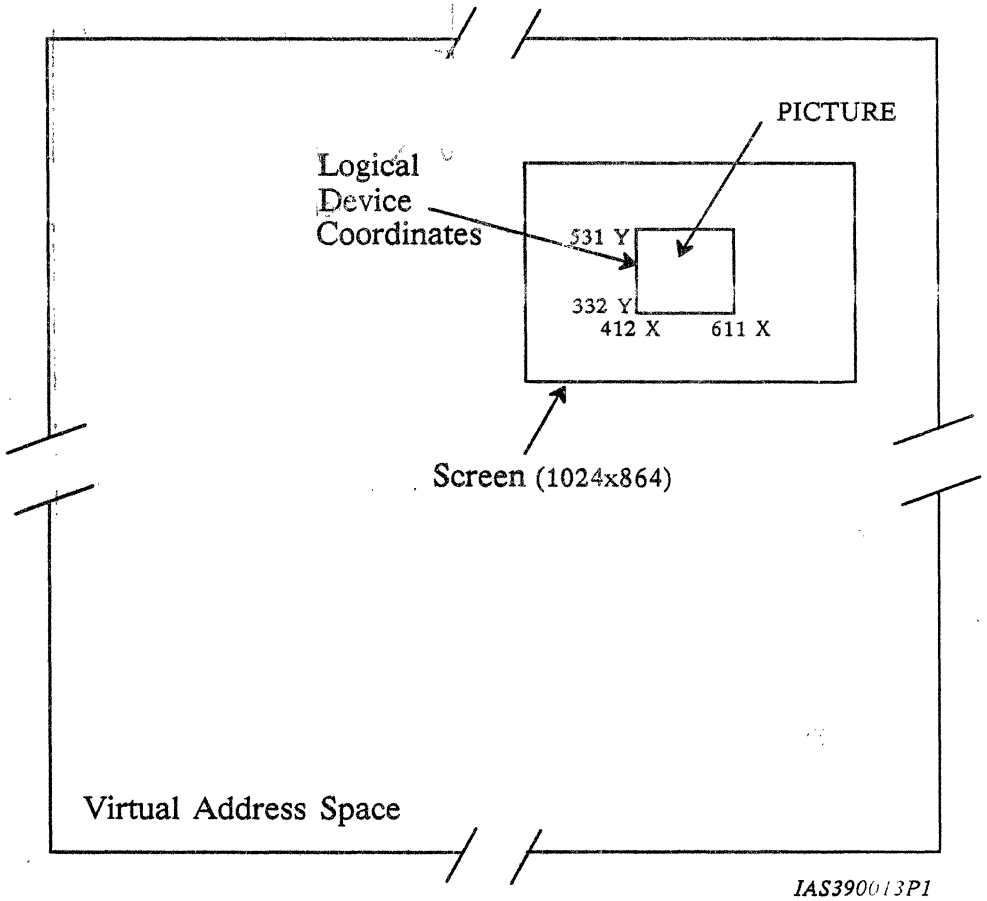


Figure 6. Centering a Raster Picture

To get the center of a 1024x1024 image on the physical screen, the logical device coordinates values should be:

```
Xmin = -0     ([1024-1024]/2)
Xmax = 1023   (xmin + 1024 - 1)
Ymin =  -80   ([864-1024]/2)
Ymax =  943   (ymin + 1024 - 1)
```

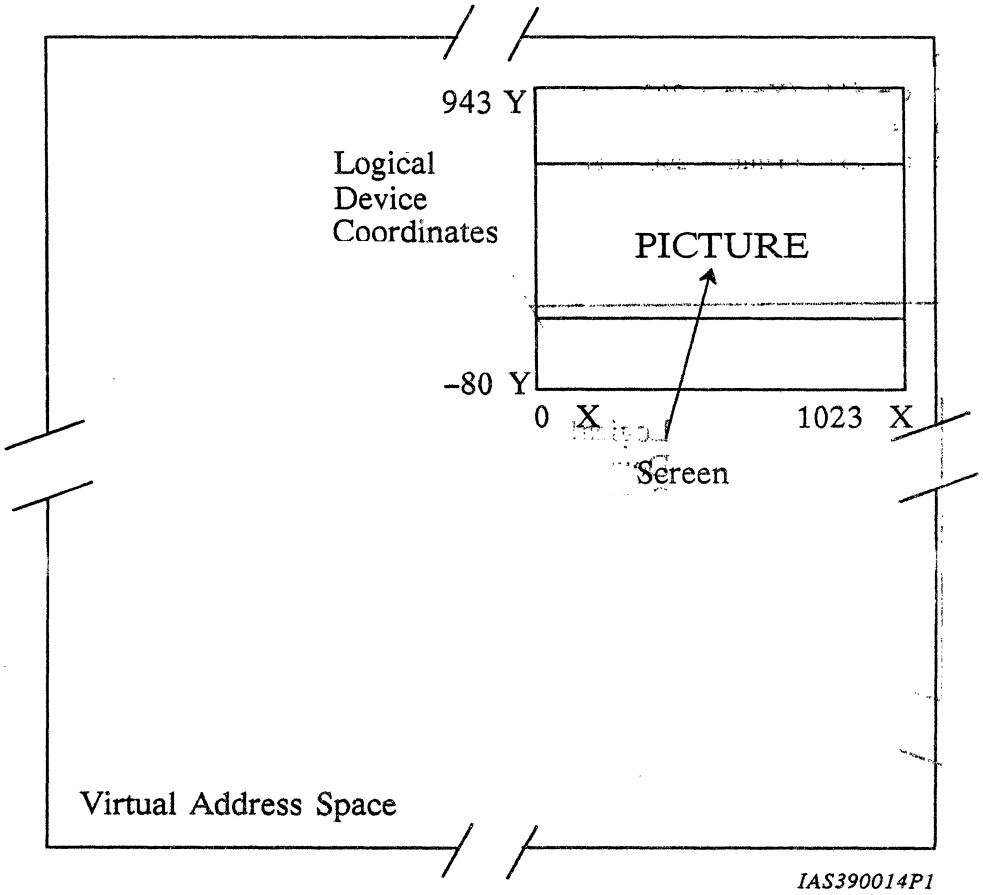


Figure 7. Displaying a Section of a Raster Picture

Encoding A Picture With Raster Modes

This section discusses the basic raster mode of the image buffer, Write Pixel Data (WRPIX), and how it functions. (Table 1 lists the basic raster mode and the results of using the mode.) The actual implementation is described in the following sections about how to use the Graphics Support Routines or user-written host routines.

The mode listed in Table 1 is described in the next section. Table 2 in the Graphics Support Routines section provides a quick reference to commands in WRPIX mode and shows the Graphics Support Routines that implement these commands.

Table 1. Raster Modes

MODE DESCRIPTION	SHORT NAME	RESULT
Write Pixel Data	WRPIX	Write red-green-blue pixel data

Writing Pixel Data (WRPIX)

There are two basic steps to encoding a picture in WRPIX mode: setting set up raster display parameters and changing the picture on the raster display.

Two WRPIX mode commands are used to establish basic operating parameters for the raster display:

- Set Logical-Device Coordinates

  This command positions the picture in virtual address space.

- Set Current Pixel Location

  This command establishes a starting point (the current pixel) in the logical device coordinates where the next WRPIX mode command begins.

Two WRPIX-mode commands are used to change the picture on the raster display:

- Erase Screen

  This command fills all of pixel memory to one value, an address into the color lookup tables.

- Load Pixel Data

  This command writes specific values to pixels.

NOTE

All data not in the range of the actual display are discarded. When the raster display is first set up in WRPIX mode, the X–Y position must be set to start position by using the X–Y position command. Data are stored in the pixels sequentially and wrap-around occurs at X maximum position until a new X–Y position is received. Whole picture representations require at least one X–Y position.

Graphics Support Routines

The Graphics Support Routines provide the easiest way to send pixel information to the display and avoid the need for writing your own routines for pixel encoding. This document assumes that you are familiar with the E&S Graphics Support Routines (GSRs). For a description of the GSRs, refer to the Graphics Support Routines user's manuals appropriate for your system and programming language in Volume 3B. The routines are listed alphabetically and, as the raster routines all begin with "PRA," they have a common location in the manuals.

Table 2 lists the mode commands, the result using the command, and the GSR calls that implement the command.

Following the table is an alphabetical list of the routines, their parameters, and a brief description.

Programming examples in FORTRAN and Pascal follow the list of the raster routines.

Table 2. Commands in WRPIX Mode

MODE COMMAND	RESULT	GSR
	Set Raster Mode	
Set Raster Mode to Write Pixel Data	Sets raster mode to write pixel data.	PRAWRP
	WRPIX Mode -- Establish Operating Parameters	
Set Logical Device Coordinates	Positions the picture in virtual address space	PRASLD
Set Current Pixel Location	Establishes the current pixel location in the LDCs where the next WRPIX mode command begins	PRASCP
	WRPIX Mode -- Change Raster Picture	
Erase Screen	Fills all of pixel memory to one address in the LUTs	PRASER
Load Pixel Data	Writes specific values to specific pixels	PRASWP

List of Raster Graphics Support Routines

The following list provides the names of the routines, expected parameters, and brief descriptions. Refer to the GSR manuals for a more detailed description.

Name of Routine and Parameters	Description
PRASCP (x,y, error routine)	Establishes current pixel location relative to the logical device coordinates.
PRASER (color, error routine	Erases the entire raster screen
PRASLD (xmin, ymin,xmax,ymax, error routine)	Sets the logical device coordinates used to position the picture in virtual address space.
PRASWP (num, pixval, error routine)	Loads current pixel location with pixel values.
PRAWRP (error routine)	Sets raster mode to write pixel data.

FORTRAN GSR Raster Programming Example

This programming example uses the Graphics Support Routines to build a "tricolor" flag display surrounded by a 20-pixel-wide blank border.

Program Example

```
C
      EXTERNAL  ERR
      INTEGER*4 MAT(4,10), BACK(3)
C
      CALL Patch ('Logdevnam=tt:/Phydevtyp=Async', Err)
C
C  ERASE SCREEN TO BLACK
C
      BACK(1) = 0
      BACK(2) = 0
      BACK(3) = 0
      CALL PRASER( BACK, ERR )
C
C  PUT ON RED RECTANGLE
C
      CALL PRASLD( 20, 20, 219, 459, ERR)
C
      MAT(1,1) = 200 * 440
      MAT(2,1) = 255
      MAT(3,1) = 0
      MAT(4,1) = 0
      CALL PRASCP( 0, 0, ERR )
      CALL PRASWP( 1, MAT, ERR )
C
C
C  PUT ON WHITE RECTANGLE
C
      CALL PRASLD( 220, 20, 419, 459, ERR)
C
      MAT(1,1) = 200 * 440
      MAT(2,1) = 255
      MAT(3,1) = 255
      MAT(4,1) = 255
```

```
      MAT(2,1) = 255
      MAT(3,1) = 255
      MAT(4,1) = 255
      CALL PRASCP( 0, 0, ERR )
      CALL PRASWP( 1, MAT, ERR )
C
C
C  PUT ON BLUE RECTANGLE
C
      CALL PRASLD( 420, 20, 619, 459, ERR)
C
      MAT(1,1) = 200 * 440
      MAT(2,1) = 0
      MAT(3,1) = 0
      MAT(4,1) = 255
      CALL PRASCP( 0, 0, ERR )
      CALL PRASWP( 1, MAT, ERR )
C
      CALL  PDTACH ( err )
      STOP
      END
```

Pascal GSR Constant Declarations

The following definitions are provided for the Pascal version of the Raster Graphics Support Routines:

```
P_MaxRunClrSize = User-specified maximum run-length color array

P_ColorType    = RECORD

      Red     : INTEGER;
      Green   : INTEGER;
      Blue    : INTEGER;
END;

P_RunColorType = RECORD

      Count   : INTEGER
      Red     : INTEGER;
      Green   : INTEGER;
      Blue    : INTEGER;
END;

P_RunClrArrayType = ARRAY [1..P_MaxRunClrSize] OF P_RunColorType;
```

Pascal GSR Raster Programming Example

```
PROGRAM EXAMPLE  (input, output);

CONST
      %INCLUDE 'PROCONST.PAS'

TYPE
      %INCLUDE 'PROTYPES.PAS'

VAR
      MAT : P_RunClrArrayType;
      BACK : P_ColorType;

      %INCLUDE 'PROEXTRN.PAS'

PROCEDURE Error_handler( err : INTEGER );

      BEGIN
      Writeln(' Error received : ', err );
      END;
```

Pascal GSR Raster Programming Example (continued)

```
BEGIN

Pattach ('Logdevnam=tt:/Phydevtyp=Async', Error_Handler );

{ ERASE SCREEN TO BLACK  }

BACK.red   := 0;
BACK.green := 0;
BACK.blue  := 0;
PRASER( BACK, Error_Handler );

{ PUT ON RED RECTANGLE  }

PRASLD( 20, 20, 219, 459, Error_Handler );

MAT[1].count := 200 * 440;
MAT[1].Red   := 255;
MAT[1].Green := 0;
MAT[1].Blue  := 0;
PRASCP( 0, 0, Error_Handler );
PRASWP( 1, MAT, Error_Handler );

{ PUT ON WHITE RECTANGLE }

PRASLD( 220, 20, 419, 459, Error_Handler );

MAT[1].count := 200 * 440;
MAT[1].Red   := 255;
MAT[1].Green := 255;
MAT[1].Blue  := 255;
PRASCP( 0, 0, Error_Handler );
PRASWP( 1, MAT, Error_Handler );

{ PUT ON BLUE RECTANGLE  }

PRASLD( 420, 20, 619, 459, Error_Handler );

MAT[1].count := 200 * 440;
MAT[1].Red   := 0;
MAT[1].Green := 0;
MAT[1].Blue  := 255;
PRASCP( 0, 0, Error_Handler );
PRASWP( 1, MAT, Error_Handler );

Pdetach ( Error_Handler );
END.
```

Formatting Raster Commands For User-generated Host Routines

Communications between the host and the PS 390 use binary data transmission protocols described in the "User's Manual for Host-Resident PS 300 I/O Subroutines" in Volume 3B and the "User Operation and Communication Guide" in Volume 1.

If you are writing your own host routines, you must:

- Ensure that the image buffer is in the correct mode (WRPIX).
- Ensure that all data (including the mode delimiter) are transferred out of the system function CIROUTE using routing byte B, which sends the data out port 21.

Mode is specified by the following decimal values:

WRPIX = 0

Raster commands are strings of data that follow this format:

0	0	0	X	X	X
Mode Delimiter		Mode		Byte Count	

Commands

The "Mode Delimiter" is two bytes of 0 (0000000000000000), and must precede a new mode specification. No delimiter needs to follow the final mode specification.

The "Mode" is the sixteen-bit binary value for WRPIX. This determines the way the data that follow are to be interpreted. If WRPIX is specified, the information that follows the byte count is interpreted as pixel data.

The "Byte Count" determines how many of the bytes of data that follow the byte count are to be interpreted as commands in the specified mode. Until a new mode delimiter is set, all data are interpreted as being in the currently specified mode. Multiple sets of byte count and data may be sent without changing modes.

The commands depend upon the specified mode issued, as described below.

WRPIX Command

0	0	0	0	XX	XX	
Mode Delimiter		WRPIX Mode		Byte Count		Mode Commands

NOTE

The maximum recommended byte count is 512 bytes.

Write Pixel Data Mode

Pixel information can be transmitted from the host to the raster display in a run-length encoding scheme.

For example, a "Load Pixel Data" command for 1-127 consecutive pixels has the following format (each character represents one bit):

```
| Onnnnnnn | RRRRRRRR | GGGGGGGG | BBBBBBBB |
+ ---------+ ---------+ ---------+ ---------+
31                                            0
```

Where "n" specifies the number of consecutive pixels, and "R-G-B" specifies the lookup table addresses for red, green, and blue.

Load Pixel Data for 128-16383 consecutive pixels command:

```
| 10nnnnnn | nnnnnnnn | RRRRRRRR | GGGGGGGG | BBBBBBBB |
+ -------- + -------- + -------- + -------- + -------- +
47                    31                              0
```

where "n" specifies the number of consecutive pixels and "R-G-B" specifies the lookup table addresses for red, green, and blue.

The current pixel location can be explicitly set by the "Set X-Y Position" command and is used to specify the current pixel location where the "Load Pixel Data" command will begin writing pixels. The pixel location is set relative to the values (Xmin, Ymin) of the logical device coordinates. If the logical device coordinates are -1024 <= X <= 1024 and -1024 <= Y <= 1024, then an X,Y position of (0,0) is the lower left-hand corner pixel and (2047,2047) is the upper right-hand corner pixel.

Set Current Pixel Location command:

```
| 110xxxxx | xxxxxxxx | 000yyyyy | yyyyyyyy |
+ -------- + -------- + -------- + -------- +
31                                          0
```

The entire pixel memory may be set to the same value with one command. This will erase the entire screen to the color in the specified lookup table locations.

Erase Screen command:

```
| 11100000 | RRRRRRRR | GGGGGGGG | BBBBBBBB |
+ -------- + -------- + -------- + -------- +
31                                          0
```

Logical device coordinates are specified by the Set Logical Device Coordinates command.

Set Logical Device Coordinates command:

```
| 11110XXX | XXXXXXXX | xxxxxxxx | xxxxxxxx | Where X = X maximum
+ -------- + -------- + -------- + -------- +       x = X minimum
  64                                  32

| 0000YYYY | YYYYYYYY | yyyyyyyy | yyyyyyyy | Where Y = Y maximum
+ -------- + -------- + -------- + -------- +       y = Y minimum
31                                          0
```

Restrictions:

```
x  =  twos complement integer     -32768    <= x <= 1023
X  =  unsigned integer                0     <= X <= 2047
y  =  twos complement integer     -32768    <= y <= 863
Y  =  unsigned integer                0     <= Y <= 2047
x <= X and y <= Y
```

Programming Example for User-Generated Host Routines

This programming example describes how to build a "tricolor" flag display surrounded by a 20-pixel-wide blank border. Numbers are specified in hexadecimal.

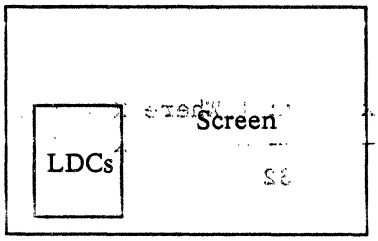
This example uses a typical set of steps required to display a raster picture. The process begins by sending a mode specification for WRLUT mode to:

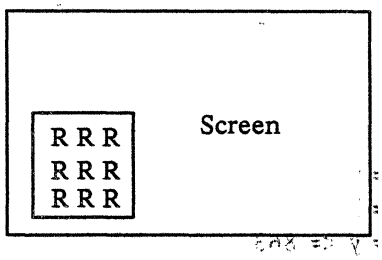
WRPIX mode to:

1. Erase screen.
2. Set Logical-Device Coordinates.
3. Set Current Pixel.
4. Load Pixel Data.

WRPIX mode (0000) is then set and 4 bytes of commands are specified. An erase screen command is sent.

```
0000            ( Set WRPIX mode )
04              ( 4 bytes of commands )
E0000000        ( Erase Screen r = 0, g = 0, b = 0 )
```

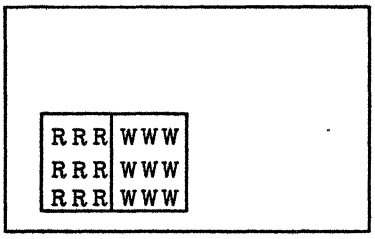





```
30              ( 48 bytes of commands )
WRPIX
F0DB001401CB0014    ( Set LDC 20 <= x <= 219, 20 <= y <= 459: Sets LDCs to
                      the left one-third of the screen )
WRPIX           ( Fill LDC area with red )
C0000000        ( Set Current Location X = 0, Y = 0 )
BFFF00FF0000    ( 16383 pixels red )
BFFF00FF0000    ( 16383 pixels red )
BFFF00FF0000    ( 16383 pixels red )
BFFF00FF0000    ( 16383 pixels red )
```

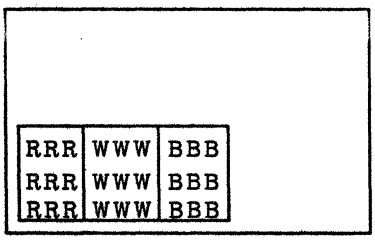
BFFF00FF0000	(16383 pixels red)
97C500FF0000	(6085 pixels red)

WRPIX (Fill new LDC with white)



30	(48 bytes of commands)
F1A300DC01CB0014	(Set LDC 220 <= x <= 419, 20 <= y <= 459)
C0000000	(Set Current Location X = 0, Y = 0)
BFFF00FFFFFF	(16383 pixels white)
BFFF00FFFFFF	(16383 pixels white)
BFFF00FFFFFF	(16383 pixels white)
BFFF00FFFFFF	(16383 pixels white)
BFFF00FFFFFF	(16383 pixels white)
97C500FFFFFF	(6085 pixels white)

WRPIX (Fill new LDC with blue)



30	(48 bytes of commands)
F26B01A401CB0014	(Set LDC 420 <= x <= 619, 20 <= y <= 459)
C0000000	(Set Current Location X = 0, Y = 0)
BFFF000000FF	(16383 pixels blue)
BFFF000000FF	(16383 pixels blue)
BFFF000000FF	(16383 pixels blue)
BFFF000000FF	(16383 pixels blue)
BFFF000000FF	(16383 pixels blue)
97C5000000FF	(6085 pixels blue)

(End of Example)

Your comments will help us provide you with more accurate, complete, and useful documentation. After making your comments in the space below, cut and fold this form as indicated, and tape to secure (please do not staple). This form may be mailed free within the United States. Thank you for your help.

Document Title _____

Document Number _____

Comments/Corrections (please include page number):

Fold

Fold

Name _____

Company and Dept. _____

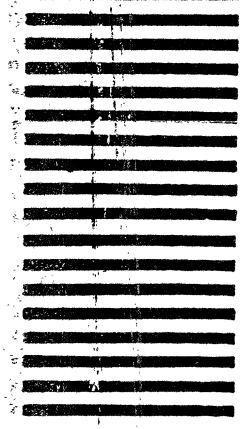
Address _____

_____

Please tape. Postal regulations prohibit stapling.

Fold

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES



BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 4432 SALT LAKE CITY, UTAH

POSTAGE WILL BE PAID BY ADDRESSEE

EVANS & SUTHERLAND
580 Arapeen Drive
Salt Lake City, Utah 84108

ATTN: IAS TECHNICAL PUBLICATIONS

Fold

Cut along dotted line.

Printed in U.S.A.