PLUTO Colour Graphics Display Controller

6, Laleham Avenue,
Mill Hill,
London NW7 3HL
(01) 959 0106

CONTENTS

1 INTRODUCTION

PLUTO is an intelligent colour graphics controller providing a display with a resolution of 640H x 288V (optionally 576V) pixels. Each pixel may be independently set to one of 8 colours.

PLUTO may be interfaced to almost any computer with very little external hardware or software. Only 2 parallel ports are required over which high level commands are sent to PLUTO. PLUTO's native bus is 80-Bus and Nasbus compatible making it easy to integrate into a customised system using readily available 8"x 8" 80-Bus processor, memory, RS-232C, IEEE 488 and I/O boards.

PLUTO uses a 16-bit 8088 microprocessor which has direct access to a large frame buffer of 192 KBytes. The frame buffer memory is dual ported so the display image may be updated at the same time as it is being refreshed onto the screen. Firmware executed by PLUTO's microprocessor performs the tasks of drawing lines and arcs, managing the display memory, colour area filling, displaying text etc. relieving the host computer of much time-consuming display manipulation. By working in parallel with the host PLUTO provides a most efficient way of sharing graphics processing.

The fast 16 bit processor and dual-ported memory combine to produce a very powerful display controller. Line drawing speeds are well in excess of 100,000 pixels/second but PLUTO's capabilities don't stop there. The built-in 'intelligence' means that arc drawing, complex shape colour filling, text display and animation are all implemented at high speed and are available using single commands from the host.

As a single board PLUTO produces an 8 colour display. An expansion bus on the board provides a simple and gradual upgrade path to an expanded system with 8 or more bits per pixel and a colour palette enabling the display of a very wide range of colour shades. PLUTO can manipulate an 8 bit-per-pixel frame buffer with the same speed and verastility as it can the 8 colour memory.

An excellent book for learning about the exciting field of computer graphics and for use as a reference is **"Principles of Interactive Computer Graphics" by William Newman and Robert Sproull** published by McGraw Hill.

2 USER'S MODEL AND TERMINOLOGY

PLUTO has a large amount of memory (the FRAME BUFFER) for storing images. The frame buffer consists of many tiny picture elements (POINTS or PIXELS) arranged in a rectangular matrix (RASTER). Each pixel may be individually assigned a number between 0 and 7 which defines its colour. To produce a viewable image a portion of the frame buffer is scanned onto a monitor screen (hence RASTER SCAN).

Three bits of memory are used to store each pixel providing 8 possible colours per pixel. The three bits can be thought of as controlling the three (light) colour components green, blue and red. The eight colours are produced by mixing these components in different combinations producing green, blue, red, cyan, yellow, magenta, white and black. If colour images are not required the frame buffer can store three monochrome pixels in the space of one colour pixel (using one bit per pixel). It is sometimes useful to consider the frame buffer as consisting of three parallel one-bit-per-pixel frame buffers glued together. These are referred to as the colour planes as there is one plane for each of the green, blue and red bits. Single planes are very useful for storing symbol or shape masks and trebles the amount of storage for such applications. Colour and monochrome images may be used together as they are stored in the same way, only the way they are interpreted is different.

The size of the frame buffer is 640 pixels wide by over 800 high and is divided into a number of smaller rectangles (PARTITIONS) each with a width of 640 pixels and each being allocated a unique identifier. Two of these partitions (partitions 1 and 2) are pre-allocated for storing two separate screen images each with a height of 288 pixels. Either of these images may be viewed on a monitor screen. An option is available that allows the two screen partitions to be displayed together providing double the vertical viewable image (ie 640H x 576V).

The remaining portion of the frame buffer is free for use as workspace or symbol storage and is allocated in partitions as required. These are called SYMBOL PARTITIONS. When used for symbol storage a symbol partition holds a number of equally sized user-specified symbols (rectangular pixel arrays). A symbol is uniquely identified by a partition identifier together with a symbol number.

One Symbol partition (partition 255) is pre-allocated and contains the 96 standard printable ASCII characters in an 8H x 10V pixel matrix. Symbols are normally numbered from 0 to n-1 where n is the number of symbols in the partition (maximum 128). The pre-defined character set is an exception to this rule where symbol numbers range from 32 to 127 to correspond with the ASCII character code.

When viewed on a monitor a partition has its origin (X=0, Y=0) at the top left corner of the screen. X co-ordinates increase from left to right, Y co-ordinates from top to bottom.

3 PLUTO'S STATE VARIABLES

PLUTO keeps a number of state variables that define the current working context. The main purpose of these is to minimise the number of parameters that are needed with each command.

When PLUTO is first powered up or after a call to PInit the state variables are initialised with useful default values. The default values are shown below in brackets by the variable name. All variables except CP are single byte quantities.

Variables marked with an asterisk (*) are only available with PLUTO's EXTENDED COMMAND firmware ROM.

3.1 COLOUR VARIABLES

A colour is a value from 0 to 255. An unexpanded PLUTO uses only 3 bits per pixel and so can only store colours 0 to 7. These appear on the screen as:

0	=	Black
1	=	Green
2	=	Blue
3	=	Cyan
4	=	Red
5	=	Yellow
6	=	Magenta
7	=	White

CCOL (7)
Current Colour. Used for most drawing and colour filling commands.

BCOL (0)
Background Colour. Used for symbol and raster copying.

FCOL (7)
Foreground Colour. Used for symbol and raster copying.

TCOL (0)
Transparent Colour. Used only with PAINT operations during raster copying.

* PCOL (7)
Perimeter (boundary) Colour. Defines the boundary for all boundary fill commands.

3.2 PARTITION AND WORKSPACE VARIABLES

CWP (1)
Current Working Partition. The CWP defines the portion of the frame buffer within which commands are to be performed. Most co-ordinates are specified relative to this partition which allows a common set of routines to be used to generate an image in various positions within the frame buffer. Appropriate values are 0 for working in a 640H x 576V area, 1 for working in screen 1 (640H x 288V) and 2 for screen 2. Other values may be used for working in user-allocated workspace partitions.

CSP (255)
Current Symbol Partition. Most symbol operations are specified relative to this partition. Facilitates a simple change of fount for example in text applications.

CDP (1)
Current Displayed Partition. Values are 0 for a 640H x 576V display, and 1 or 2 for one of the 640H x 288V partitions.

CP (0,0)
Current Position as an X,Y co-ordinate pair. Most commands use this as a starting point and update it after completion to the most useful position for subsequent commands. CP is set explicitly by one of the Move commands.

3.3 MISCELLANEOUS VARIABLES

STATUS (0)
Records the result of the most recent command. Zero means that the operation was successful, other values are error codes (refer to section 5 for interpretation).

* PAT (240 = 0f0 hex)
The binary pattern optionally used for line and arc commands.

3.4 SINGLE COLOUR PLANE OPERATORS

WPROT (0)
Write Protect mask. This 8 bit number uses one bit per colour plane to selectively protect planes from being modified. In an 8 colour system bit 0 (least significant bit) is used to protect the green plane, bit 1 the blue plane and bit 2 the red plane (bits 3 to 7 are not used). Setting a bit protects the corresponding plane. WPROT is only used for Copy, LImagC and LsymC commands.

* RSEL (7)
Read Select mask. This is the converse of WPROT and selects colour planes for use as the source of an operation. RSEL has the effect of converting a monochrome (one-bit-per-pixel) image stored in a single colour plane into a two colour image and has its main use with text and symbol display. Each plane can hold a different set of symbols increasing the total symbol storage space threefold.

As with WPROT an 8 bit mask is used on a bit-per-plane basis to select planes from which to read an image. RSEL is normally used to select a single plane but may be used to select multiple planes. Pixels in the selected plane(s) are converted to either the foreground (FCOL) or background (BCOL) colour before being used in an operation. The pixel value read from the frame buffer is logically ANDed with RSEL - if the result is non-zero FCOL is used otherwise BCOL is used. An appropriate value for RSEL to read an image from the green plane for example would be 1.

3.5 STYLE (80h=128)

The STYLE variable modifies the effect of most commands. Using a single variable in this way minimises the amount of setting up required before using a command. Although the variety of STYLEs may at first seem overwhelming the power of the STYLE parameter will quickly become realised. The default STYLE produces sensible results from all functions so its value need not be set until special effects are required.

STYLE is interpreted differently by different classes of commands and these classes will be described separately. STYLE is a byte variable (ie it can have any value from 0 to 255). The definition is given below in terms of bit fields for those familiar with working in binary and also as decimal-weighted fields.

3.5.1 LINE AND ARC STYLES

For lines and arcs only the least significant 4 bits of STYLE are used, the 4 most significant bits may be any value. STYLE may be split into the following bit fields:

```
   7     6     5     4     3     2     1     0
:=====:=====:=====:=====:=====:=====:=====:======:
:  X  :  X  :  X  :  X  :  P  :   FUNC    :  SK  :
:=====:=====:=====:=====:=====:=====:=====:======:
```

	FUNC	MEANING	WEIGHTING HEX(DEC)
	RPL	Replace points on line/arc with CCOL	0 (0)
	XOR	Logical XOR points on line/arc with CCOL	2 (2)
	OR	Logical OR points on line/arc with CCOL	4 (4)
	AND	Logical AND points on line/arc with CCOL	6 (6)

	P	MEANING	WEIGHTING HEX(DEC)
*	Pattern	Apply pattern PAT while drawing line/arc	8 (8)
		Draw continuous line/arc	0 (0)

	SK	MEANING	
*	Skip	Don't draw first point on line/arc	1 (1)
		Draw all points on line/arc	0 (0)

* NOTE
P and SK are only available with PLUTO's EXTENDED COMMAND firmware ROM.

To select the appropriate value of STYLE add the values in the WEIGHTING column corresponding to the facilities required. For example to draw a line using the XOR function using a pattern and without plotting the first point on the line a value of 2+8+1=11 is used.

The pattern mask is used as follows: the next point on the line or arc is calculated; if the top bit of the pattern mask is set then the point is plotted using CCOL and FUNC otherwise no action is taken; the pattern mask is rotated left by one bit and the operation repeated for all other points on the line or arc.

The Skip bit suppresses the first point on the line or arc. This is useful when for example the XOR function is used while drawing connected lines since the last point of one line, being the same as the first point of the following line, would be plotted twice causing gaps to appear in the lines. It is also useful for plotting complete circles for a similar reason.

As an example of using FUNC, if a line were drawn using XOR with a colour CCOL of white (7), each pixel along the line would be colour complemented. Although the WPROT flag is not effective for lines and arcs the effect of write protection may be achieved with appropriate use of colour and style. For example, to draw a line only in the blue colour plane a STYLE of OR (4) and CCOL of blue (2) would be used. To erase a line in the blue plane a CCOL of yellow (5) (the complement of blue) and a STYLE of AND would be used.

3.5.2 RECTANGLE FILL AND PLOT STYLES

For Rfill (rectangle fill) and Plot commands STYLE has the following format:

```
      7     6     5     4     3     2     1     0
   :=====:=====:=====:=====:=====:=====:=====:=====:
   :  X  :  X  :  X  :  X  :  X  :    FUNC   :  X  :
   :=====:=====:=====:=====:=====:=====:=====:=====:
```

FUNC	MEANING	WEIGHTING HEX(DEC)
RPL	Fill the rectangle with colour CCOL	0 (0)
XOR	XOR each pixel in the rectangle with CCOL	2 (2)
OR	OR each pixel in the rectangle with CCOL	4 (4)
AND	AND each pixel in the rectangle with CCOL	6 (6)

Rfill is a very fast function for operating on a rectangle using a constant colour and operator. Particularly useful operations are RPL for filling the area with a constant colour and XOR with white (CCOL=7) for colour complementing the area for highlighting. Although the write protect facility cannot be directly applied with Rfill the same effect can be achieved with appropriate combinations of FUNC and CCOL, for example FUNC=AND with CCOL=1 clears all but the least significant plane.

Plot commands affect a single pixel. The pixel is combined with CCOL using one of the functions above. For example RPL sets the pixel to CCOL.

3.5.3 COPY STYLES

The definition of STYLE for functions CopyS (copy symbol), Copy rectangle and CopyTS (copy to symbol) is:

```
      7     6     5     4     3     2     1     0
   :=====:=====:=====:=====:=====:=====:=====:=====:
   :  RS :  WP :    ROT    :       FUNC      :  N  :
   :=====:=====:=====:=====:=====:=====:=====:=====:
```

FUNC	MEANING	WEIGHTING HEX	(DEC)
RPL	Copy source raster to destination	0	(0)
XOR	Logical XOR source raster with destination	2	(2)
OR	Logical OR source raster with destination	4	(4)
AND	Logical AND source raster with destination	6	(6)
PAINT	Paint source raster onto destination	8	(8)

N	MEANING		
NOT	Logically invert source before operation	1	(1)
	Use source without inversion	0	(0)

ROT	MEANING		
0	Don't rotate source raster	0	(0)
90	Rotate source 90 degrees before operation	10	(16)
180	Rotate source 180 degrees before operation	20	(32)
270	Rotate source 270 degrees before operation	30	(48)

WP	MEANING		
Use WPROT	Write protect planes specified by WPROT	40	(64)
	Don't write protect any planes	0	(0)

RS	MEANING		
Use RSEL	Use only FCOL and BCOL in destination	80	(128)
	Perform operation normally	0	(0)

These three functions use a common implementation within PLUTO's firmware but offer convenient shorthand methods of specifying common operations (displaying and saving symbols). They all use a general raster (rectangle) copying operation known as RasterOp. RasterOp takes two rasters of equal size, a source and a destination, and replaces the destination raster with a function of the source and destination. The operation may be any of the FUNCs listed above and is applied between each pixel in the source raster and its corresponding pixel in the destination. The logical operations are performed on all bits (planes) of the pixel simultaneously.

RPL can be used for scrolling parts of the display using a destination raster that is horizontally alligned with and above the source. A one line vertical spacing produces a smooth scroll while a 10 line spacing produces a character line scroll. RPL is also the appropriate function for defining a symbol that is already on the screen using CopyTS.

XOR is useful for making non-destructive changes to the display. An example is superimposing a cursor on an image. The cursor (eg a symbol) is XOR'd onto the image to display it and then XOR'd again to remove it.

NOT combined with any function is used to invert (ie colour complement) the source raster before it is used. Using the same raster for both source and destination and an RPL function, for example, can be used to highlight an area (although the Rfill command may be used to achieve this effect more efficiently).

PAINT produces the effect of copying non-rectangular shapes. The source and destination are defined exactly as for other RasterOps but pixels of a chosen colour are not copied to the destination. The chosen colour is defined by TCOL (the 'transparent' colour). For example, a symbol may be defined as a coloured shape on a black background and TCOL set to black. The function is so-called as it is useful in painting programs using brushes defined as symbol shapes.

ROT rotates the source raster anticlockwise through a multiple of 90 degrees before performing the operation (but doesn't change the source in the frame buffer). Note that while overlapping rasters are always copied non-destructively when no rotation is specified it is impossible to always guarantee a non-destructive copy with rotation.

WP invokes the WPROT (write protect) flag during a RasterOp which protects selected planes from being modified (see section 3.4). A situation where this is useful is when superimposing a text display on a graphics background using one plane for the text and the remaining planes for graphics. To scroll the text the graphics planes are write protected and an RPL RasterOp used.

RS has its main use with text and symbol display. The colours in the source raster are converted to either the foreground (FCOL) or background (BCOL) colour before the operation is performed (see section 3.4). This means that once symbols are defined they may be displayed in any foreground and background colour combinations by simply changing FCOL and BCOL.

PLUTO's RasterOps are extremely flexible enabling most required effects to be achieved. In spite of this flexibility PLUTO's firmware is optimised so that simple functions such as RPL are performed at maximum possible speed.

4 PLUTO'S USER-CALLABLE FUNCTIONS

4.1 INTRODUCTION

This section defines the commands built into PLUTO's firmware. Commands are classified in the definitions below as FUNCTIONS or PROCEDURES the difference being that FUNCTIONs return one or more result values while PROCEDURES don't. A FUNCTION or PROCEDURE may have zero or more parameters, the format for each being PROCEDURE(<parameter list>) and FUNCTION(<parameter list>):<return value(s)> Each FUNCTION or PROCEDURE has a unique code that is sent to PLUTO to invoke the command following which parameters are sent in left to right order as defined below. All values sent to or received from PLUTO are 8 bit quantities. Refer to section 7 for detailed information on the protocol for invoking functions and reading results.

Most commands use one or more of PLUTO's state variables to minimise the number of parameters that need to be supplied with each call, one of the most useful of which is the current position pointer CP. This is updated after most operations to the position that would be most useful for subsequent commands, for example after drawing a line the

CP is positioned at the end of the line so that a sequence of line commands will draw connected lines without the need to explicitly position the CP. There will always be situations of course where CP will not be appropriate and must be changed using a Move command.

Some commands use co-ordinates relative to the CP and enable position-independent images to be created. Image-drawing software routines using these commands can arrange to make the sum of all X movements zero and all Y movements zero so the final CP is the same as the initial CP, facilitating the production of modular and transportable software.

The value of all of PLUTO's state variables may be set and inquired. The ability to read the current setting of these variables is important in a multi-process environment enabling a working context to be inquired and saved by an interrupting process which temporarily takes over control of the display and then restored to its original state.

4.2 COMMAND DEFINITIONS

KEY
Upper case variables are 16 bit quantities, lower case 8 bits. All values are passed between PLUTO and the host 8 bits at a time. All co-ordinates are relative to the current working partition (CWP).

X	X co-ordinate (a 16 bit unsigned value)
Y	Y co-ordinate (a 16 bit unsigned value)
xl	Least significant 8 bits of X
xh	Most significant 8 bits of X
yl	Least significant 8 bits of Y
yh	Most significant 8 bits of Y
DX	X increment (a 16 bit signed value)
DY	Y increment (a 16 bit signed value)
dxl	Least significant 8 bits of DX
dxh	Most significant 8 bits of DX
dyl	Least significant 8 bits of DY
dyh	Most significant 8 bits of DY
dx	Short X increment (an 8 bit signed value)
dy	Short Y increment (an 8 bit signed value)
W	Width (a 16 bit unsigned value)
H	Height (a 16 bit unsigned value)
wl	Least significant 8 bits of W
wh	Most significant 8 bits of W
n	A general 8 bit unsigned value
c	Colour
p	Partition identifier

Because each parameter must be an 8 bit quantity a 16 bit quantity is sent as two 8 bit values. For example to move the current position pointer by a relative amount DX=+5, DY=-2 (the MoveR command) the distances are first expressed in two's complement form (DX+5, DY=0fffe hex) and are then split into two halves giving dxl=5, dxh=0, dyl=0fe hex, dyh=0ff hex.

Commands labelled with an asterisk (*) are only available in PLUTO's EXTENDED COMMAND firmware ROM.

Commands labelled with a plus (+) are only available with the DOUBLE RESOLUTION (640H x 576V) option.

4.2.1 HOUSEKEEPING COMMANDS

PROCEDURE PInit
Initialises PLUTO. All state variables are set to their default values, symbol partitions freed and the frame buffer cleared to all black.

PROCEDURE ClrCWP
Clears the current working partition (sets all pixels within it to zero). The CP is not modified.

FUNCTION AllocP(wl,wh,hl,hh,n):p
Allocates a symbol or workspace partition. W and H are the width and height of each symbol and n is the number of symbols required. To reserve a workspace partition W,H would be the size of the required partition and n would be 1. The maximum number of symbols per partition is 128 (80 hex) and the maximum number of user partitions is 8. Symbols are numbered from 0 to n-1. The value returned from AllocP is either 255 (0ff hex) if the function failed to allocate a partition or else a partition identifier (for use in subsequent commands, eg SCSP). If the function returns an error code result the reason for failure can be found by using IStat. Partitions cannot be deleted.

+ PROCEDURE SHires
Produces a screen resolution of 640H x 576V using an interlaced field display. CDP is automatically set to 0 for consistency.

+ PROCEDURE SLores
Produces a standard display resolution of 640H x 288V (reverses the effect of SHires). If CDP is currently 0 then it is set to 1 for consistency. If the CP has a Y greater then 287 then it is set to 0. Calls to SHires and SLores may be used freely.

PROCEDURE SCCol(c)
Sets the current working colour to c. This is used for line and arc drawing, raster and area filling and point plotting.

PROCEDURE SFCol(c)
Sets the foreground colour to c.

PROCEDURE SBCol(c)
Sets the background colour to c.

PROCEDURE STCol(c)
Sets the transparent colour used in PAINT operations (in PChar, Copy and CopyTS) to c.

PROCEDURE SStyle(n)
Sets the style for subsequent commands to n.

PROCEDURE SCDP(p)
Sets the current display partition to p (must be 0, 1 or 2 for double resolution, screen 1 or screen 2 respectively). The CDP should only be explicitly set to 1 or 2 to display one of the two 640H x 288V screen images. It is automatically set to 0 by SHires and restored to 1 by SLores to enable the user to determine which screen resolution is being used (via ICDP).

PROCEDURE SCWP(p)
Sets the current working partition to p. This establishes the size and position of a section of PLUTO's memory to which all co-ordinates are related. The value of p will normally be 0 for working within the double sized display (640H x 576V) or 1 or 2 for one of the two images of 640H x 288V. User-allocated partitions may also be used for the CWP in which case the partition may be freely used for general workspace instead of storing symbols. Most commands operate within the CWP.

PROCEDURE SCSP(p)
Sets the current symbol partition to p. The CSP may be set to 255 (0ff hex) to use the default character set or to any user-allocated partition number (allocated by a call to AllocP).

PROCEDURE SWprot(n)
Sets the write protect mask to n. WPROT is used only for selected commands and if the WP bit in STYLE is set. WPROT uses one bit per colour plane to selectively prevent changes being made within the plane (changes are prevented if the bit is set).

* PROCEDURE SRsel(n)
Sets the read select mask to n. RSEL is used only for selected commands and if the RS bit in STYLE is set. RSEL uses one bit per colour plane to select a plane (or planes) to use in the operation.

* PROCEDURE SPat(n)
Sets the pattern mask to n. PAT is used for line and arc drawing if the P bit in STYLE is set.

* PROCEDURE SPCol(c)
Sets the perimeter colour to c. PCOL is used in boundary fills to define the border colour of the area to be filled.

FUNCTION ICCol:c
Returns the current value of CCOL.

FUNCTION IFCol:c
Returns the current value of FCOL.

FUNCTION IBCol:c
Returns the current value of BCOL.

FUNCTION ITCol:c
Returns the current value of TCOL.

FUNCTION IStyle:n
Returns the current value of STYLE.

FUNCTION ICDP:p
Returns the current value of CDP.

FUNCTION ICWP:p
Returns the current value of CWP.

FUNCTION ICSP:p
Returns the current value of CSP.

FUNCTION IWprot:n
Returns the current value of WPROT.

* FUNCTION IRSel:n
Returns the current value of RSEL.

* FUNCTION IPat:n
Returns the current value of PAT.

* FUNCTION IPCol:c
Returns the current value of PCOL.

FUNCTION IStat:n
Returns the current value of STATUS. STATUS records the result of the last command and has a value of 0 for a successful operation or an error code that explains the reason for failure of the command. A list of error codes can be found in section 5.

4.2.2 CURRENT POSITION COMMANDS

PROCEDURE MoveTo(xl,xh,yl,yh)
Moves the CP to X,Y relative to the current working partition. If the co-ordinate lies outside the partition then CP is not changed.

PROCEDURE MoveR(dxl,dxh,dyl,dyh)
Moves the CP by a relative amount DX,DY. Otherwise as for MoveTo.

PROCEDURE MoveRS(dx,dy)
Moves the CP by a relative amount dx,dy. Otherwise as for MoveTo. Useful for short movements (+127/-128) as only two parameters are required.

FUNCTION ICP:xl,xh,yl,yh
Returns the current position CP.

4.2.3 LINE DRAWING COMMANDS

PROCEDURE LineTo(xl,xh,yl,yh)
Draws a line from CP to X,Y. The appearance of the line is controlled by the current STYLE (see section 3.5.1). The CP is updated to point to the end of the line (ie the last point that was plotted). If X,Y lies outside of the current working partition then none of the line is drawn and the CP is not modified.

PROCEDURE LineR(dxl,dxh,dyl,dyh)
Draws a line from CP to a point DX,DY relative to CP. Otherwise as for LineTo.

PROCEDURE LineRS(dx,dy)
Draws a line from CP to a point dx,dy relative to CP. Otherwise as for LineTo. Useful for short lines (end point +127/-128 from CP).

4.2.4 ARC AND CIRCLE COMMANDS

* PROCEDURE Arc(dxcl,dxch,dycl,dych,dxel,dxeh,dyel,dyeh)
Draws an arc from the CP in an anti-clockwise direction to a point DXE,DYE relative to CP using a very fast incremental algorithm. The arc has a centre of curvature at a point DXC,DYC relative to CP. The centre of curvature needn't lie inside the current working partition permitting an arc with a large curvature to be drawn. By selecting negative or positive values for DXC and DYC as appropriate an arc may be drawn with any direction of curvature. If the specified end point doesn't lie on the arc then the arc will stop at the nearest point on the path of the arc. If the end point is grossly in error then the arc will stop on a quadrant boundary.

To cope with the two possible viewable resolutions (640H x 288V and 640H x 576V) which have different aspect ratios the arc is plotted asymmetrically by scaling the Y co-ordinates to make the arc appear as a segment of a circle. The aspect ratio is chosen according to the current working partition. When working in partition 0 a 640H x 576V resolution is assumed and one screen unit in the Y direction is equivalent to 0.87 of a unit in the X direction otherwise a 640H x 288V resolution is assumed where one screen unit in Y is equivalent to 1.66 units in X. In both cases the space between adjacent pixels in the X direction is taken as the unit of measurement. This should be taken into account when calculating XC,YC and XE,YE which are screen co-ordinates. See below for examples. The range of values for XC (and YC before scaling) is +/-1024.

The arc command differs from all others in that it may be only partially executed if it extends outside of the current working partition. In this case the command is terminated when the arc reaches the partition boundary.

The arc is plotted according to the setting of STYLE (see section 3.5.1) and the CP is moved to the last point on the arc.

EXAMPLE 1
Draw an arc from the current position in working partition 1 with radius 1000 units and ends subtending angles of 30 degrees and 35 degrees to the horizontal.

SOLUTION
If Dx and Dy are the distances respectively in the X and Y directions from the centre of curvature of the arc to the CP then

    Dx1 = Radius*cos(30)
    Dy1 = a*Radius*sin(30)   where a = 1.66 for partition 1

For an arc in the first quadrant DXC=-Dx1, YXC=Dy1 (remembering that Y co-ordinates increase from the top of the screen to the bottom).

Similarly for the distance from the centre to the end point:

    Dx2 = Radius*cos(35)
    Dy2 = a*Radius*sin(35)
and
    DXE = Dx2 - DXC
    DYE = -Dy2 + DYC
This gives
    DXC = -1000 * 0.866   = -866
    DYC = 1/1.66 * 1000 * 0.5   = 301
    DXE = 1000 * 0.819 - 866   = -47
    DYE = -1/1.66 * 1000 * 0.574 + 301 = -45
    (all values rounded to the nearest integer)

Using Hex notation:
    DXC = 0FC9E
    DYC =   12D
    DXE = 0FFD1
    DYE = 0FFD3

Hence the command parameters are: Arc(9e,0fc,2d,1,0d1,0ff,0d3,0ff).

EXAMPLE 2
Draw a circle in partition one with centre at (200,100) and radius 50 units.

SOLUTION
Move the CP to (200+Radius,100) = (250,100). The start and end points are both at the CP. The centre and end points are:

    DXC = -Radius
        = -50
    DYC = 0
    DXE = 0
    DYE = 0

Hence the command parameters are Arc(0ce,0ff,0,0,0,0,0,0) (using hex notation).

4.2.5 AREA FILL COMMANDS

PLUTO features an extensive set of fast, powerful routines for filling any general closed area with a solid colour or pattern. The pattern area fills greatly ease the production of information displays and open up possibilies that would otherwise be impossible. Painting programs can produce composite colours and textured patterns by using subtle combinations of colours in the pattern. Graphs and bar charts may be constructed and shaded with meaningful keyed patterns. These commands above all show PLUTO's versatility and speed.

PROCEDURE Rfill(wl,wh,hl,hh)
Fills a rectangle of width W and height H whose top left corner is at the CP with colour CCOL. Various combining functions may be used between the raster and CCOL as defined by STYLE (see section 3.5.2). The CP is incremented by W in the X direction. If the new X overflows the screen width then it is zeroed and the CP is incremented by H in the Y direction. If the new Y overflows the partition height then it is zeroed. If the raster lies partially outside the CWP then no operation is performed.

PROCEDURE Ffill
Flood fills any arbitrary closed area with colour CCOL. The colour of the pixel at the CP defines the colour of the area to be filled. The area is made up of this pixel and all connected pixels of the same colour. If the area is not enclosed completely by pixels of a contrasting colour then the filling colour will leak out and ultimately stop at the partition boundary. Pixels outside of the CWP are never modified and the CP is not moved.

* PROCEDURE Bfill
Boundary fills any arbitrary closed area with colour CCOL. This differs from Ffill in that the filled area is defined by the boundary colour defined by PCOL (perimeter colour). The filled area starts at the CP and grows out in all directions until the boundary colour is met. For this command to operate correctly pixels in the area to be filled must not have colour CCOL (unless this is the same as PCOL).

This function would be used instead of Ffill to wipe a restricted area containing more than one colour.

* PROCEDURE FfilS
Flood fills a simple closed area (eg a circle or triangle). A similar command to Ffill but is optimised for simple areas which are distinguished by not having internal holes or any interior angle of more than 180 degrees. Some shapes with obtuse internal angles can be coped with if the large angles are in the 'sides' (for example an hour-glass shape is acceptable).

The advantage of using FfilS is its three-fold speed improvement over the already fast Ffill command. Inconsiderate use of FfilS on convoluted areas may cause the shape of the filled area to be destroyed.

* PROCEDURE BfilS
Boundary fills a simple closed area. Performs a similar function to Bfill but with the limitations on the filled area that FfilS imposes.

* PROCEDURE FfilP
Flood fills any arbitrary closed shape with a user-defined pattern. The algorithm used for defining the area is the same as that used for Ffill but a pattern instead of a constantt colour is used to fill the area. The pattern is 'wallpapered' into the area so that two similarly filled adjacent areas are combined they merge together invisibly.

The pattern is set up before FfilP is invoked and may be defined in one of two ways - either as a symbol or an arbitrary rectangle. In either case the pattern may be of arbitrary size (see below). The RSEL mask may be applied to the pattern to alter the foreground and background colours of the pattern and to enable a pattern to be stored on a single colour plane for space economy. The RS bit of STYLE enables the use of RSEL - if this is not set then the pattern is copied directly from the source (refer to section 3 for a full discussion).

To ensure correct operation of FfilP the colour of the area being filled must not appear in the pattern. If this condition is violated then FfilP may potentially continue indefinitely. This is prevented by a timeout which guarantees to stop the command which may only fill part of the area.

There is no corresponding BfilP command. This is because the restriction with boundary fills that the fill pattern mustn't contain any of the colours that appear within the area to be filled limits the use of such a function to very few situations.

* PROCEDURE FfilSP
Flood fills a simple closed area with a pattern. The filled area is defined using the method described for for FfilS and the operation performed is as described for FfilP. This increases still further the speed of pattern filling if the area is known to be simple.

* PROCEDURE BfilSP
Boundary fills a simple area with a pattern. The filled area is defined using the method described for for BfilS and the operation performed is as described for FfilP.

* PROCEDURE SFpatS(p,n)
Sets the fill pattern for all pattern fills to be a symbol. The symbol may be user-defined or one of the characters from the built-in character fount. The command parameters are the symbol partition identifier p and the symbol number n.

* PROCEDURE SfPatR(wl,wh,hl,hh,p,xl,xh,yl,yh)
Sets the fill pattern for all pattern fills to be an arbitrary rectangle. The rectangle may be in any partition and of any size up to the maximum size of the partition. The parameters to SfPatR define the rectangle width W and height H, the partition p in which it lies and the co-ordinates X,Y of its top left corner. This function fixes only

the position at which the pattern is to be found - the image within this rectangle may be freely changed to produce different patterns.

A wide range of effects can be achieved using this function. A text string pattern may be used to write inside an arbitrary area; a single sloping line to produce a striped shading pattern and a pair of crossed lines to produce a cross-hatch shading.

4.2.6 RASTER COPYING COMMANDS

The three raster copy commands are implemented using a common set of routines within PLUTO but take different parameters for convenience in use.

Very often an image is created using line, arc and area fill primitives. Once the image has created the raster moving commands provide a very efficient and flexible method of manipulating it. Refer to section 3.5.3 for a full discussion of these commands.

PROCEDURE Copy(wl,wh,hl,hh,p1,x1l,x1h,y1l,y1h,p2,x2l,x2h,y2l,y2h)
This is the general form of the raster copy command. The source and destination rasters are each of width W and height H (if a rotation of 90 or 270 degrees is specified then the destination raster has a width of H and height of W). The source raster is in partition p1 with top left corner at X1,Y1, while the destination is in partition p2 at X2,Y2. This command doesn't use the CWP or CP (and doesn't modify either) providing a general method of copying images between partitions. The effect of the copy operation depends on the current STYLE. If any of the parameters are out of range then no operation is performed.

PROCEDURE CopyS(n)
Copies a symbol. The source raster is symbol number n within the current symbol partition whose width and height are already known to PLUTO. The destination raster is at the CP within the current working partition. After the operation the CP is updated using the method describe for Rfill (section 4.2.5). This command is useful for writing text using either the default character set or a user-defined fount.

This command differs from all others in that only the parameter is sent to PLUTO (ie there is no CopyS command code). All symbol numbers are between 0 and 127 (7f hex) inclusive and command codes are between 128 and 255 (80h and 0ffh). When a value less than 128 is sent to PLUTO it is interpreted as a symbol number and the command CopyS is automatically invoked with this as its parameter. This allows text to be used very efficiently needing only one code to be sent per character (for example sending pluto a command code of 41 hex prints the character 'A' if the current symbol partition is 255).

PROCEDURE CopyTS(n)
Copies a raster to a symbol. The source raster is at the CP in the current working partition. The destination raster is the place where symbol number n in the current symbol partition is stored. The size of the raster is the size of the symbol.

This command is one way of defining a symbol. To produce an exact copy of the source, appropriate for multi-coloured symbols, the STYLE would be set to 0. Where symbols are drawn using only two colours (foreground and background) and symbol storage space is at a premium the symbol may be stored in a single colour plane. To do this the WPROT mask is set to protect al but the desired destination plane and RSEL to select the plane on which the source symbol is defined. FCOL should be set to 7 and BCOL 0 to and the WP and RS bits in STYLE enabled. Refer to section 3 for a full explanation of WPROT and RSEL.

4.2.7 SINGLE PIXEL COMMANDS

These commands allow the image to be operated upon very flexibly for situations where none of the higher level primitives perform the desired effect for example to trace out irregular outlines. Wherever possible the higher level commands should be used to minimise the number of command calls and increase efficiency.

PROCEDURE Plot(xl,xh,yl,yh)
Plots a point at X,Y using colour CCOL. The effect is modified by the current STYLE (see section 3.5.2). The CP is moved to X,Y.

PROCEDURE PlotR(dxl,dxh,dyl,dyh)
Plots a point at DX,DY relative to the CP. Otherwise as for Plot.

PROCEDURE PlotRS(dx,dy)
Plots a point at dx,dy relative to the CP. Otherwise as for Plot. Useful for short movements (+127/-128 from CP).

FUNCTION RPix(xl,xh,yl,yh):c
Returns the colour of the pixel at X,Y. The CP is moved to X,Y.

FUNCTION RPixR(dxl,dxh,dyl,dyh):c
Returns the colour of the pixel at a point DX,DY relative to the CP. The CP is adjusted by DX,DY.

FUNCTION RPixRS(dx,dy):c
Returns the colour of the pixel at a point dx,dy relative to the CP. The CP is adjusted by dx,dy (maximum relative movement +127/-128 in either direction).

4.2.8 IMAGE AND SYMBOL READ AND LOAD COMMANDS

These commands permit reading and loading the whole or part of a partition directly and are useful for saving images on backing storage via the host and for defining symbol shapes. A choice of data formats is provided to cater for full colour one-byte-per-pixel or compressed two-colour images. The compressed forms increase the amount of image and symbol storage by using the frame buffer as three single bit planes. They also provide a more compact form for data transfer to and from the host improving transfer rate and decreasing the external storage space required.

PROCEDURE LImage(wl,wh,hl,hh,<W x H pixels>)
Transfers a sequence of pixels into a specified raster. The raster has its top left corner at the CP relative to the current working partition and width W and height H. Each byte of data defines the colour of one pixel. Pixels are transferred to the raster in the order that a TV picture is scanned - starting with the top left corner, moving left to right W pixels then starting at the left edge of the next row. The correct number of pixels must be supplied. The CP is moved by W units in the X direction (as for Rfill section 4.2.5).

PROCEDURE LSym(n,<W x H pixels>)
Transfers a sequence of pixels into symbol n of the CSP (to define a symbol). PLUTO knows the size of the symbol so this is not specified in the parameter list. LSym is used in the same way as LImage.

FUNCTION RImage(wl,wh,hl,hh):<W x H pixels>
The converse of LImage. After sending the command the host must read W x H bytes, one byte per pixel in the raster. The raster is defined and read in the way described for LImage.

FUNCTION RSym(n):<W x H pixels>
The converse of LSym used for retrieving symbol definitions from the frame buffer.

PROCEDURE LImagC(wl,wh,hl,hh,<data bytes>)
Loads an image from compressed data into selected colour planes. Each data byte defines the colour of 8 consecutive pixels in the raster being loaded, moving from left to right starting with the the most significant bit. If a bit in the data byte is set (1) the foreground colour (FCOL) is written to the corresponding pixel otherwise the background colour (BCOL) is written. The WPROT flag is used to load selected planes. For example to load the image into the green plane only FCOL is chosen to be 1, BCOL to be 0 and WPROT to be 6. The command parameters define the width and height of the raster as for LImage. If the width is not an exact multiple of 8 then it is rounded up to the nearest multiple of 8 and divided by 8 to determine the number of data bytes to be sent for each row.

PROCEDURE LSymC(n,<data bytes>)
Loads a symbol from compressed data. LSymC works in the same way as LImagC and is useful for storing symbols which are defined using a bit-per-point mask on selected planes effectively trebling the amount of symbol storage space. The symbol is normally converted into selected foreground and background colours when it is used.

FUNCTION RImagC(wl,wh,hl,hh):<data bytes>
Reads a raster and compresses the data into 8 pixels per data byte. The converse of LImagC, used mainly to read an image stored on a single plane. The plane is selected using RSEL (see section 3.4).

FUNCTION RSymC(n):<data bytes>
Reads a symbol and compresses the data. The converse of LSymC.

5. ERROR RECOVERY AND ERROR CODES

If inappropriate command parameters are sent to PLUTO or for some other reason the command cannot be executed PLUTO's state variable STATUS is assigned a value to reflect the reason for failure. This value can be read using the IStat command. Possible error codes are:

CODE	MEANING
1	The co-ordinates used for a command were outside of the partition boundary. This could arise from width or height values being too large or absolute or relative offsets producing out of bounds co-ordinate values.
2	Command code not implemented.
3	Partition identifier used with the command has not been allocated.
4	No more free partitions available (in response to an AllocP call).
5	Width or height parameters for an AllocP call are zero or too large.
9	A symbol number supplied to the command was greater than the number of symbols in the partition.
10	Illegal partition number for a call to SCDP.
11	No more space for symbol partitions.
12	Rotation was specified for a CopyTS command (not allowed).
13	FfilP was given an impossible area to pattern fill and 'timed out'.
14	An arc that crossed the partition boundary was clipped.
15	Parameters to an arc command were too big.

6 PLUTO INSTALLATION

PLUTO is built on an 80-Bus and Nasbus compatible 8" x 8" board and will plug directly into either type of bus. Section 6.1 describes the installation into such a system.

PLUTO is designed to be very easily interfaced to almost any computer with a minimum of external components. Section 6.3 provides details of PLUTO's interface requirements.

6.1 INSTALLATION INTO 80-BUS AND NASBUS SYSTEMS

Only two I/O ports are required for communication with PLUTO. The ports have consecutive addresses that may be selected to be on any 20 hex byte boundary. PLUTO decodes 4 addresses two of which are not used but reserved for future use.

TO SELECT THE I/O ADDRESS link the point marked COM next to IC30 to one of the points 0 to 7 as required:

Base address (hex)	Link COM to
00	4
20	5
40	6
60	7
80	0
A0	1
C0	2
E0	3

PLUTO is pre-configured with a base address of A0.

For compatibility with Nascom systems a NASIO signal is optionally provided by PLUTO. Only one board in the entire system should provide this signal which is asserted when an I/O address for the Nascom main board is decoded. If this signal is to be provided by PLUTO then the points marked NASIO should be linked. PLUTO asserts this signal for all I/O addresses from 0 to 07f hex inclusive which means that all peripheral boards (including PLUTO) should use I/O address above 80 hex. The Nascom Internal/External I/O addressing switch must be set to enable external addressing.

For compatibility with Nascom 1 systems a DBDR option is provided by linking the points marked.

The points marked WAIT should not be linked for any system.

6.2 CONNECTING AND SELECTING A VIDEO MONITOR

The connector marked PLA provides the video output signals which are suitable for connection to a standard RGB colour monitor.

Separate horizontal and vertical sync pulses are provided as well as a combined (horizontal and vertical) sync pulse to cater for most types of monitor. All syncs are negative TTL level pulses and the three colour signals (Red, Green and Blue) are positive TTL levels. Alternate pins on PLA are grounded for maximum noise immunity.

A composite video signal is provided on the green channel for use with a standard monochrome monitor (1V video level).

To make full use of PLUTO's high resolution display a so-called 'high resolution' RGB colour monitor is preferred. Such monitors have a dot spacing of .31mm or less and are capable of displaying a 640H x 288V resolution. There is no harm in using a medium or low resolution colour monitor, only the picture quality will suffer. A standard horizontal scan rate of 15.625 KHz and vertical frame rate of 50Hz is used.

For the double resolution display of 640H x 576V PLUTO produces an interlaced display. Most standard monitors are capable of working with such a display.

The connections to PLA are shown below. Pin 1 is marked with a small arrow and pins are numbered as for standard 20 way ribbon connectors.

PLA pin	Signal
1	GREEN
3	GROUND
5	HORIZONTAL SYNC
7	BLUE
9	GROUND
11	VERTICAL SYNC
13	COMBINED HORIZ. AND VERT. SYNCS
15	COMPOSITE VIDEO
17	GROUND
19	RED
2,4,6,8,10,12,14,16,18,20	GROUND

6.3 HARDWARE INTERFACING DETAILS FOR NON-80 BUS SYSTEMS

This section lists PLUTO's signal and timing requirements. Numbers in brackets are the pin numbers on PLUTO's 78 way bus.

ADDRESS LINES A2-A4 (32-34)
These should be at logic 0 level to select PLUTO.

ADDRESS LINES A5-A7 (35-37)
These address PLUTO according to the setting of the address decode link as explained in section 6.1. For example if COM is linked to address decode 4 then A5-A7 should be at logic 0 level to select PLUTO.

ADDRESS LINE A0 (30)
Selects between the COMMAND (A0 logic level 1) and STATUS (A0 logic level 0) ports.

DB0-DB7 (50-57)
Bi-directional data bus.

/IORQ (26)
This should be at logic level 0 to select PLUTO (it may be kept permanently at logic 0).

/RD (29)
Read strobe (active low) to enable data from PLUTO onto the data bus. Data access time from the leading edge (high-to-low) of /RD is 70nsec max. The address and /IORQ lines should be stable 30 nsecs prior to /RD leading edge and stay valid for 40 nsecs after the trailing edge.

/WR (28)
Write strobe (active low) to write data from from the data bus to PLUTO. Data is latched on the trailing edge (low-to-high transition) of /WR. Data need not be valid before the leading edge of /WR but must be valid 30 nsecs before the trailing edge and remain valid for 40 nsecs after. The address and /IORQ lines should be stable 30 nsecs

prior to /WR leading edge and stay valid until 40 nsecs after the trailing edge.

+5v (75,76,77,78)
A single +5 volt supply is required. Typical current consumption is 1.5 Amps.

GROUND (1,2,3,4)

All other signals are 80-Bus specific and are not required by PLUTO.

7 GENERAL PROGRAMMING NOTES

Two I/O ports are used for communication with PLUTO. One of these is a bi-directional data port (the COMMAND port) over which all commands and parameters are sent and results received. The second port is used as a STATUS port when read by the host and as a reset port when written. The port addresses are slectable (see section 6.1). In some computer installations PLUTO may be memory-mapped in which case two memory locations are used.

Commands are sent to PLUTO's COMMAND port as a series of bytes - the command code followed by the command parameters in left-to-right order. PLUTO cannot accept a command if it is processing a previous one so a check must be made to see whether PLUTO is ready to accept data by reading the STATUS port. Only the top bit of the status value is important - it is set (ie value >= 128) if PLUTO is ready to accept a data byte and zero otherwise.

The STATUS port may be read at any time without affecting PLUTO's operation. To avoid the necessity of checking the STATUS port before sending every byte of data PLUTO guarantees certain behaviour characteristics. To gain the optimum performance from PLUTO the following rules should be followed when sending a command:

1. Read and test the value returned from the STATUS port. If the value is greater than 127 (top bit set) then go on to step 2 otherwise repeat step 1.

2. Send the first byte of data (the command code) to the COMMAND port.

3. Wait for PLUTO to decode the command by testing the STATUS port as described in step 1.

4. Send the remaining data bytes (the parameters) without checking the STATUS port if the maximum transfer rate is 4 microseconds per byte (2.5 microseconds with a PLUTO operating on an 8MHz clock). There is no minimum speed at which data must be sent. If the host is a Z80 micro running at 4MHz, for example PLUTO's maximum data rate cannot be exceeded.

For commands that return one or more data bytes PLUTO must be in a ready state before the first result byte is read from the COMMAND port. Thereafter result bytes may be read at a maximum data rate of 4 microseconds per byte (2.5 microseconds for an 8MHz PLUTO) without the need to check the STATUS port.

There are a few commands that cannot transfer data at the maximum rate mentioned above, namely the read and load data commands (LSym, LImage, RSym and RImage) and the compressed data read and load commands (LSymC, LImagC, RSymC and RImagC). With the normal read and load commands pixel data may be transferred at the maximum stated rate while each horizontal line in the raster is being transferred but the STATUS port must be checked between lines. For the compressed read and load commands it is advisable to check the STATUS for each byte transferred.

It is important to send the correct number of parameters with a command and to read the correct number of result bytes. If this is not done then PLUTO may misinterpret subsequent commands. A command may be terminated during parameter or result transfer by sending 4 zero value bytes to the STATUS port (when writing, the STATUS port becomes a RESET port). The STATUS port must be checked first by reading it to make sure that PLUTO is ready to accept the reset command. The STATUS need only be checked before sending the first zero byte. The RESET will cause PLUTO to re-ready itself for the beginning of a new command which may be sent when the STATUS port returns to the ready state.

APPENDIX A: COMMAND CODE SUMMARY

KEY:

Upper case variables are 16 bit quantities, lower case 8 bits. All values passed between PLUTO and the host are 8 bits. All co-ordinates are relative to the current working partition (CWP).

X	X co-ordinate (a 16 bit unsigned value)
Y	Y co-ordinate (a 16 bit unsigned value)
xl	Least significant 8 bits of X
xh	Most significant 8 bits of X
yl	Least significant 8 bits of Y
yh	Most significant 8 bits of Y
DX	X increment (a 16 bit signed value)
DY	Y increment (a 16 bit signed value)
dxl	Least significant 8 bits of DX
dxh	Most significant 8 bits of DX
dyl	Least significant 8 bits of DY
dyh	Most significant 8 bits of DY
dx	Short X increment (an 8 bit signed value)
dy	Short Y increment (an 8 bit signed value)
W	Width (a 16 bit unsigned value)
H	Height (a 16 bit unsigned value)
wl	Least significant 8 bits of W
wh	Most significant 8 bits of W
n	General 8 bit number
p	Partition identifier

HEX	DEC	COMMAND	PARAMETERS	RETURN VALUES
A3	163	AllocP	wl,wh,hl,hh,n	Partition number
C1	193	Arc	xcl,xch,ycl,ych, xel,xeh,yel,yeh	
B0	176	Bfill		
B1	177	BfilS		
B3	179	BfilSP		
AC	172	ClrCWP		
85	133	Copy	wl,wh,hl,hh, p1,x1l,x1h,y1l,y1h p2,x2l,x2h,y2l,y2h	
0-7F	0-127	CopyS		
84	132	CopyTS	Symbol number	
82	130	Ffill		
AE	174	FfilP		
AD	173	FfilS		
AF	175	FfilSP		
90	144	IBCol		Background colour
8D	141	ICCol		Current colour
96	150	ICP		xl,xh,yl,yh
94	148	ICDP		Display partition
92	146	ICSP		Symbol partition
93	147	ICWP		Working partition

HEX	DEC	COMMAND	PARAMETERS	RETURN VALUES
8F	143	IFCol		Foreground colour
B7	183	IPat		Pattern
BB	187	IPCol		Colour
B9	185	IRsel		Read select mask
86	134	IStat		Status
8E	142	IStyle		Style
91	145	ITCol		Transparent colour
95	149	IWprot		Write protect mask
9F	159	LImage	wl,wh,hl,hh, <WxH pixels>	
BC	188	LImagC	wl,wh,hl,hh, <packed pixels>	
9D	157	LineR	dxl,dxh,dyl,dyh	
9E	158	LineRS	dx,dy	
80	128	LineTo	xl,xh,yl,yh	
A0	160	LSym	n,<WxH pixels>	
BD	189	LSymC	n,<packed pixels>	
98	152	MoveR	dxl,dxh,dyl,dyh	
99	153	MoveRS	dx,dy	
97	151	MoveTo	xl,xh,yl,yh	
A6	166	PInit		
9A	154	Plot	xl,xh,yl,yh	
9B	155	PlotR	dxl,dxh,dyl,dyh	
9C	156	PlotRS	dx,dy	
81	129	RFill	wl,wh,hl,hh	
A1	161	RImage	wl,wh,hl,hh	W x H pixels
BE	190	RImagC	wl,wh,hl,hh	Packed pixels
A7	167	RPix	xl,xh,yl,yh	Colour
A8	168	RPixR	dxl,dxh,dyl,dyh	Colour
A9	169	RPixRS	dx,dy	Colour
A2	162	RSym	Symbol number	W x H pixels
BF	191	RSymC	Symbol number	Packed pixels
87	135	SBCol	New background colour	
89	137	SCCol	New current colour	
83	131	SCDP	New display partition	
A4	164	SCSP	New symbol partition	
A5	165	SCWP	New working partition	
88	136	SFCol	New foreground colour	
B4	180	SFPatR	wl,wh,hl,hh, p,xl,xh,yl,yh	
B5	181	SFPatS	p,n	
AA	170	SHires		
AB	171	SLores		
B6	182	SPat	New Pattern	
BA	186	SPCol	New boundary colour	
B8	184	SRsel	New read select mask	
8A	138	SStyle	New style	
8B	139	STCol	New transparent colour	
8C	140	SWprot	New write protect mask	

```
                      ; PLUTO Z80 DEMO         ***   L.J.NOBLE     ***
                      ;
                      ; IO RESEARCH LTD.       ***   01 959 0106   ***
                      ;
0100                  START  ORG  100H
0100 CD0703                  CALL MAIN
0103                         END
                      ;
                      ; PORT DEFS
          =00A1       DATA   EQU 0A1H
          =00A0       STATUS EQU 0A0H
                      ;
                      ; COMMAND CODES
                      ;
          =00A3       ALLOC$ EQU 0A3H
          =0085       COPY$  EQU 085H
          =008D       ICCOL$ EQU 08DH
          =0096       ICP$   EQU 096H
          =0093       ICWP$  EQU 093H
          =00A6       INIT$  EQU 0A6H
          =009D       LREL$  EQU 09DH
          =009E       LRELS$ EQU 09EH
          =0080       LTO$   EQU 080H
          =00A0       LSYM$  EQU 0A0H
          =0098       MREL$  EQU 098H
          =0099       MRELS$ EQU 099H
          =0097       MTO$   EQU 097H
          =0082       PFILL$ EQU 082H
          =0081       RFILL$ EQU 081H
          =0087       SBCOL$ EQU 087H
          =0089       SCCOL$ EQU 089H
          =0088       SFCOL$ EQU 088H
          =00A4       SCSP$  EQU 0A4H
          =00A5       SCWP$  EQU 0A5H
          =0083       SCDP$  EQU 083H
          =008A       SSTLE$ EQU 08AH
          =009C       PRS$   EQU 09CH
          =009A       PAT$   EQU 09AH
                      ;
                      ; COLOURS
                      ;
          =0004       RED    EQU 4
          =0002       BLUE   EQU 2
          =0001       GREEN  EQU 1
          =0005       YELLOW EQU 5
          =0003       CYAN   EQU 3
          =0006       MAGENT EQU 6
          =0000       BLACK  EQU 0
          =0007       WHITE  EQU 7
          =0008       PAINT  EQU 08H   ;PAINT STYLE VAL
                      ;
                      ; VARIABLES
                      ;
0103 00               C1     DEFB 0
0104 00               COL    DEFB 0
0105 0000             DX     DEFW 0
```

```
0107 0000   DY      DEFW 0
0109 0000   SX      DEFW 0
010B 0000   SY      DEFW 0
010D 0000   WIDTH   DEFW 0
010F 0000   HEIGHT  DEFW 0
0111 00     PSRC    DEFB 0
0112 00     PDST    DEFB 0
0113 00     TIM1    DEFB 0
0114 00     TIM2    DEFB 0
0115 00     COUNT   DEFB 0
0116 0000   INC     DEFW 0
0118 00     COLL    DEFB 0
0119 0000   X1      DEFW 0
011B 0000   Y1      DEFW 0

            ;
            ;
            ;
            ; PRIMITIVES
            ;
011D F5     READY   PUSH AF
011E DBA0   RDY     IN   A,(STATUS)
0120 E680           AND  80H
0122 28FA           JR   Z,RDY
0124 F1             POP  AF
0125 C9             RET
            ;
0126 CD1D01 INIT    CALL READY
0129 3E00           LD   A,0
012B D3A0           OUT  (STATUS),A
012D D3A0           OUT  (STATUS),A
012F D3A0           OUT  (STATUS),A
0131 D3A0           OUT  (STATUS),A
0133 CD1D01         CALL READY
0136 3EA6           LD   A,INIT$
0138 D3A1           OUT  (DATA),A
013A C9             RET
            ;
013B 0EA1   PLOTAT  LD   C,DATA
013D CD1D01         CALL READY
0140 3E9A           LD   A,PAT$
0142 ED79           OUT  (C),A
0144 CD1D01         CALL READY
0147 ED69           OUT  (C),L
0149 ED61           OUT  (C),H
014B ED59           OUT  (C),E
014D ED51           OUT  (C),D
014F C9             RET
            ;
0150 0EA1   PLOTRS  LD   C,DATA
0152 CD1D01         CALL READY
0155 3E9C           LD   A,PRS$
0157 ED79           OUT  (C),A
0159 CD1D01         CALL READY
015C ED59           OUT  (C),E
015E ED51           OUT  (C),D
0160 C9             RET
            ;
```

```
0161 0EA1     MOVETO LD    C,DATA
0163 CD1D01          CALL  READY
0166 3E97            LD    A,MTO$
0168 ED79            OUT   (C),A
016A CD1D01          CALL  READY
016D ED69            OUT   (C),L
016F ED61            OUT   (C),H
0171 ED59            OUT   (C),E
0173 ED51            OUT   (C),D
0175 C9              RET
              ;
0176 0EA1     MOVREL LD    C,DATA
0178 CD1D01          CALL  READY
017B 3E98            LD    A,MREL$
017D ED79            OUT   (C),A
017F CD1D01          CALL  READY
0182 ED69            OUT   (C),L
0184 ED61            OUT   (C),H
0186 ED59            OUT   (C),E
0188 ED51            OUT   (C),D
018A C9              RET
              ;
018B 0EA1     MOVRLS LD    C,DATA
018D CD1D01          CALL  READY
0190 3E99            LD    A,MRELS$
0192 ED79            OUT   (C),A
0194 CD1D01          CALL  READY
0197 ED51            OUT   (C),D
0199 ED59            OUT   (C),E
019B C9              RET
              ;
019C 0EA1     SCCOL  LD    C,DATA
019E CD1D01          CALL  READY
01A1 3E89            LD    A,SCCOL$
01A3 ED79            OUT   (C),A
01A5 CD1D01          CALL  READY
01A8 ED61            OUT   (C),H
01AA C9              RET
              ;
01AB 0EA1     SFCOL  LD    C,DATA
01AD CD1D01          CALL  READY
01B0 3E88            LD    A,SFCOL$
01B2 ED79            OUT   (C),A
01B4 CD1D01          CALL  READY
01B7 ED61            OUT   (C),H
01B9 C9              RET
              ;
01BA 0EA1     SSTLE  LD    C,DATA
01BC CD1D01          CALL  READY
01BF 3E8A            LD    A,SSTLE$
01C1 ED79            OUT   (C),A
01C3 CD1D01          CALL  READY
01C6 ED61            OUT   (C),H
01C8 C9              RET
              ;
01C9 0EA1     SBCOL  LD    C,DATA
01CB CD1D01          CALL  READY
01CE 3E87            LD    A,SBCOL$
```

```
01D0 ED79            OUT   (C),A
01D2 CD1D01          CALL  READY
01D5 ED61            OUT   (C),H
01D7 C9              RET
                ;
01D8 0EA1       SETCSP LD    C,DATA
01DA CD1D01          CALL  READY
01DD 3EA4            LD    A,SCSP$
01DF ED79            OUT   (C),A
01E1 CD1D01          CALL  READY
01E4 ED61            OUT   (C),H
01E6 C9              RET
                ;
01E7 0EA1       SETCWP LD    C,DATA
01E9 CD1D01          CALL  READY
01EC 3EA5            LD    A,SCWP$
01EE ED79            OUT   (C),A
01F0 CD1D01          CALL  READY
01F3 ED61            OUT   (C),H
01F5 C9              RET
                ;
01F6 0EA1       SETCDP LD    C,DATA
01F8 CD1D01          CALL  READY
01FB 3E83            LD    A,SCDP$
01FD ED79            OUT   (C),A
01FF CD1D01          CALL  READY
0202 ED61            OUT   (C),H
0204 C9              RET
                ;
0205 0EA1       LTO    LD    C,DATA
0207 CD1D01          CALL  READY
020A 3E80            LD    A,LTO$
020C ED79            OUT   (C),A
020E CD1D01          CALL  READY
0211 ED69            OUT   (C),L
0213 ED61            OUT   (C),H
0215 ED59            OUT   (C),E
0217 ED51            OUT   (C),D
0219 C9              RET
                ;
021A 0EA1       LREL   LD    C,DATA
021C CD1D01          CALL  READY
021F 3E9D            LD    A,LREL$
0221 ED79            OUT   (C),A
0223 CD1D01          CALL  READY
0226 ED69            OUT   (C),L
0228 ED61            OUT   (C),H
022A ED59            OUT   (C),E
022C ED51            OUT   (C),D
022E C9              RET
                ;
022F 0EA1       LRELS  LD    C,DATA
0231 CD1D01          CALL  READY
0234 3E9E            LD    A,LRELS$
0236 ED79            OUT   (C),A
0238 CD1D01          CALL  READY
023B ED51            OUT   (C),D
023D ED59            OUT   (C),E
```

```
023F C9                     RET
                    ;
0240 CD1D01         PFILL   CALL READY
0243 3E82                   LD   A,PFILL$
0245 D3A1                   OUT  (DATA),A
0247 C9                     RET
                    ;
0248 0EA1           RFILL   LD   C,DATA
024A CD1D01                 CALL READY
024D 3E81                   LD   A,RFILL$
024F ED79                   OUT  (C),A
0251 CD1D01                 CALL READY
0254 ED69                   OUT  (C),L
0256 ED61                   OUT  (C),H
0258 ED59                   OUT  (C),E
025A ED51                   OUT  (C),D
025C C9                     RET
                    ;
025D CD1D01         WRITE   CALL READY
0260 0EA1                   LD   C,DATA
0262 ED61                   OUT  (C),H
0264 C9                     RET
                    ;
0265 7E             PRINT   LD   A,(HL)
0266 B7                     OR   A
0267 C8                     RET  Z
0268 CD1D01                 CALL READY
026B D3A1                   OUT  (DATA),A
026D 23                     INC  HL
026E 18F5                   JR   PRINT
                    ;
0270 0EA1           ICP     LD   C,DATA
0272 CD1D01                 CALL READY
0275 3E96                   LD   A,ICP$
0277 D3A1                   OUT  (DATA),A
0279 CD1D01                 CALL READY
027C ED68                   IN   L,(C)
027E ED60                   IN   H,(C)
0280 ED58                   IN   E,(C)
0282 ED50                   IN   D,(C)
0284 C9                     RET
                    ;
0285 0EA1           ICWP    LD   C,DATA
0287 CD1D01                 CALL READY
028A 3E93                   LD   A,ICWP$
028C D3A1                   OUT  (DATA),A
028E CD1D01                 CALL READY
0291 DBA1                   IN   A,(DATA)
0293 C9                     RET
                    ;
0294 0EA1           COPY    LD   C,DATA
0296 CD1D01                 CALL READY
0299 3E85                   LD   A,COPY$
029B ED79                   OUT  (C),A
029D CD1D01                 CALL READY
02A0 2A0D01                 LD   HL,(WIDTH)
02A3 ED69                   OUT  (C),L
02A5 ED61                   OUT  (C),H
```

```
02A7 2A0F01            LD    HL,(HEIGHT)
02AA ED69              OUT   (C),L
02AC ED61              OUT   (C),H
02AE 3A1101            LD    A,(PSRC)
02B1 ED79              OUT   (C),A
02B3 2A0901            LD    HL,(SX)
02B6 ED69              OUT   (C),L
02B8 ED61              OUT   (C),H
02BA 2A0B01            LD    HL,(SY)
02BD ED69              OUT   (C),L
02BF ED61              OUT   (C),H
02C1 3A1201            LD    A,(PDST)
02C4 ED79              OUT   (C),A
02C6 2A0501            LD    HL,(DX)
02C9 ED69              OUT   (C),L
02CB ED61              OUT   (C),H
02CD 2A0701            LD    HL,(DY)
02D0 ED69              OUT   (C),L
02D2 ED61              OUT   (C),H
02D4 C9                RET
                ;
                ;
                ; DELAY
                ;
02D5 3EFF       TDEL   LD    A,255
02D7 321301     T1     LD    (TIM1),A
02DA 321401     T2     LD    (TIM2),A
02DD 06FF       T2A    LD    B,255
02DF 10FE       T3     DJNZ  T3
02E1 3A1401            LD    A,(TIM2)
02E4 3D                DEC   A
02E5 20F3              JR    NZ,T2
02E7 3A1301            LD    A,(TIM1)
02EA 3D                DEC   A
02EB 2001              JR    NZ,T4
02ED C9                RET
02EE 321301     T4     LD    (TIM1),A
02F1 18EA              JR    T2A
                ;
                ;
02F3 7E         SHAPE1 LD    A,(HL)
02F4 B7                OR    A
02F5 C8                RET   Z  ;FINISHED SHAPE
02F6 1805              JR    SHAPE2
                ;
02F8 7E         SHAPE  LD    A,(HL)
02F9 23                INC   HL
02FA B7                OR    A
02FB 28F6              JR    Z,SHAPE1   ;0 FOUND
02FD 5E         SHAPE2 LD    E,(HL)
02FE 2B                DEC   HL
02FF 56                LD    D,(HL)
0300 CD2F02            CALL  LRELS
0303 23                INC   HL
0304 23                INC   HL
0305 18F1              JR    SHAPE
                ;
                ;
```

```
                   ; *** MAIN PROGRAM ***
                   ;
0307 CD2601    MAIN    CALL INIT
030A 2604              LD   H,RED
030C CD9C01            CALL SCCOL
030F CD4002            CALL PFILL
                   ;
                   ; DRAW PLUTO
                   ;
0312 3E02              LD   A,2
0314 320301    PLUTO   LD   (C1),A
0317 2602              LD   H,2
0319 CDBA01            CALL SSTLE ;XOR
031C 3E0B              LD   A,11
031E 321501    LOOP    LD   (COUNT),A
0321 211900            LD   HL,25
0324 111400            LD   DE,20
0327 CD6101            CALL MOVETO
032A 0608              LD   B,8  ;COUNT
032C 219106    LOOP1   LD   HL,DAT1
032F CDF802            CALL SHAPE
0332 161E              LD   D,30
0334 1E0A              LD   E,10
0336 CD8B01            CALL MOVRLS
0339 219F06            LD   HL,DAT2
033C CDF802            CALL SHAPE
033F 165A              LD   D,90
0341 1EF6              LD   E,-10
0343 CD8B01            CALL MOVRLS
0346 21A906            LD   HL,DAT3
0349 CDF802            CALL SHAPE
034C 1678              LD   D,120
034E 1E00              LD   E,0
0350 CD8B01            CALL MOVRLS
0353 21B706            LD   HL,DAT4
0356 CDF802            CALL SHAPE
0359 1678              LD   D,120
035B 1E00              LD   E,0
035D CD8B01            CALL MOVRLS
0360 21C906            LD   HL,DAT5
0363 CDF802            CALL SHAPE
0366 1678              LD   D,120
0368 1E00              LD   E,0
036A CD8B01            CALL MOVRLS
036D 21DB06            LD   HL,DAT6
0370 CDF802            CALL SHAPE
0373 161E              LD   D,30
0375 1E14              LD   E,20
0377 CD8B01            CALL MOVRLS
037A 21E506            LD   HL,DAT7
037D CDF802            CALL SHAPE
0380 1688              LD   D,-120
0382 1EED              LD   E,-19
0384 CD8B01            CALL MOVRLS
0387 1E00              LD   E,0
0389 CD8B01            CALL MOVRLS
038C CD8B01            CALL MOVRLS
038F CD8B01            CALL MOVRLS
```

```
0392 16E4              LD   D,-28
0394 CD8B01            CALL MOVRLS
0397 1093              DJNZ LOOP1
                ;
0399 3A1501            LD   A,(COUNT)
039C 3D                DEC  A
039D C21E03            JP   NZ,LOOP
                ;
                ; FILL IN
                ;
03A0 160B              LD   D,11
03A2 1E08              LD   E,8
03A4 3E07              LD   A,7
03A6 320401     LOOP2  LD   (COL),A
03A9 67                LD   H,A
03AA CD9C01            CALL SCCOL
03AD CD8B01            CALL MOVRLS
03B0 0605              LD   B,5
03B2 CD4002     LOOP3  CALL PFILL
03B5 1678              LD   D,120
03B7 1E00              LD   E,0
03B9 CD8B01            CALL MOVRLS
03BC 10F4              DJNZ LOOP3
                ;
03BE 1688              LD   D,-120   ;MOVE TO 'P'
03C0 1E00              LD   E,0
03C2 CD8B01            CALL MOVRLS
03C5 CD8B01            CALL MOVRLS
03C8 CD8B01            CALL MOVRLS
03CB CD8B01            CALL MOVRLS
03CE 3A0401            LD   A,(COL)
03D1 3D                DEC  A
03D2 20D2              JR   NZ,LOOP2
                ;
03D4 CDA505            CALL LDEL
                ;
03D7 2600              LD   H,0
03D9 CDBA01            CALL SSTLE
03DC 210500            LD   HL,5
03DF 110500            LD   DE,5
03E2 CD6101            CALL MOVETO
03E5 2602              LD   H,BLUE
03E7 CD9C01            CALL SCCOL
03EA CD4002            CALL PFILL
03ED 2600              LD   H,BLACK
03EF CD9C01            CALL SCCOL
03F2 217602            LD   HL,630
03F5 111601            LD   DE,278
03F8 CD4802            CALL RFILL
03FB 2602              LD   H,BLUE
03FD CD9C01            CALL SCCOL
0400 CD4802            CALL RFILL
0403 2604              LD   H,RED
0405 CD9C01            CALL SCCOL
0408 CDD502            CALL TDEL
040B 3A0301            LD   A,(C1)
040E 3D                DEC  A
040F C21403            JP   NZ,PLUTO
```

```
0412 C31804            JP    FLAG
                 ;
                 ; RANDOM NUMBER
0415 ED5F        RAND  LD    A,R
0417 C9                RET
                 ;
                 ; FLAG
                 ;
0418 210000      FLAG  LD    HL,0
041B 110000            LD    DE,0
041E CD6101            CALL  MOVETO
0421 2602              LD    H,BLUE
0423 CD9C01            CALL  SCCOL
0426 218002            LD    HL,640
0429 112001            LD    DE,288
042C CD4802            CALL  RFILL     ;BLUE BACK
042F 2607              LD    H,WHITE
0431 CD9C01            CALL  SCCOL
0434 213200            LD    HL,50
0437 110000            LD    DE,0
043A CD6101            CALL  MOVETO
043D 217F02            LD    HL,639
0440 110C01            LD    DE,268
0443 CD0502            CALL  LTO
0446 210000            LD    HL,0
0449 111400            LD    DE,20
044C CD6101            CALL  MOVETO
044F 214E02            LD    HL,590
0452 111F01            LD    DE,287
0455 CD0502            CALL  LTO
0458 210000            LD    HL,0
045B 110C01            LD    DE,268
045E CD6101            CALL  MOVETO
0461 214E02            LD    HL,590
0464 110000            LD    DE,0
0467 CD0502            CALL  LTO
046A 213200            LD    HL,50
046D 111F01            LD    DE,287
0470 CD6101            CALL  MOVETO
0473 217F02            LD    HL,639
0476 111400            LD    DE,20
0479 CD0502            CALL  LTO
047C 210000            LD    HL,0
047F 110000            LD    DE,0
0482 CD6101            CALL  MOVETO
0485 CD4002            CALL  PFILL
0488 217F02            LD    HL,639
048B 110000            LD    DE,0
048E CD6101            CALL  MOVETO
0491 CD4002            CALL  PFILL
0494 217E02            LD    HL,638
0497 111F01            LD    DE,287
049A CD6101            CALL  MOVETO
049D CD4002            CALL  PFILL
04A0 210000            LD    HL,0
04A3 111F01            LD    DE,287
04A6 CD6101            CALL  MOVETO
04A9 CD4002            CALL  PFILL
```

```
04AC 2604           LD    H,RED
04AE CD9C01         CALL  SCCOL
04B1 210000         LD    HL,0
04B4 110000         LD    DE,0
04B7 CD6101         CALL  MOVETO
04BA 217F02         LD    HL,639
04BD 111F01         LD    DE,287
04C0 CD0502         CALL  LTO
04C3 210000         LD    HL,0
04C6 110E00         LD    DE,14
04C9 CD6101         CALL  MOVETO
04CC 214001         LD    HL,320
04CF 119F00         LD    DE,159
04D2 CD0502         CALL  LTO
04D5 210000         LD    HL,0
04D8 111F01         LD    DE,287
04DB CD6101         CALL  MOVETO
04DE 217F02         LD    HL,639
04E1 110000         LD    DE,0
04E4 CD0502         CALL  LTO
04E7 212300         LD    HL,35
04EA 111F01         LD    DE,287
04ED CD6101         CALL  MOVETO
04F0 214001         LD    HL,320
04F3 119F00         LD    DE,159
04F6 CD0502         CALL  LTO
04F9 215C02         LD    HL,604
04FC 110000         LD    DE,0
04FF CD6101         CALL  MOVETO
0502 214001         LD    HL,320
0505 118000         LD    DE,128
0508 CD0502         CALL  LTO
050B 217F02         LD    HL,639
050E 111101         LD    DE,273
0511 CD6101         CALL  MOVETO
0514 214001         LD    HL,320
0517 118000         LD    DE,128
051A CD0502         CALL  LTO
051D 210000         LD    HL,0
0520 110200         LD    DE,2
0523 CD6101         CALL  MOVETO
0526 CD4002         CALL  PFILL
0529 217D02         LD    HL,637
052C 110000         LD    DE,0
052F CD6101         CALL  MOVETO
0532 CD4002         CALL  PFILL
0535 217F02         LD    HL,639
0538 111D01         LD    DE,285
053B CD6101         CALL  MOVETO
053E CD4002         CALL  PFILL
0541 210200         LD    HL,2
0544 111F01         LD    DE,287
0547 CD6101         CALL  MOVETO
054A CD4002         CALL  PFILL
054D 2607           LD    H,WHITE
054F CD9C01         CALL  SCCOL
0552 210701         LD    HL,263
0555 110000         LD    DE,0
```

```
055B CD6101           CALL MOVETO
055B 217300           LD   HL,115
055E 111F01           LD   DE,287
0561 CD4802           CALL RFILL ;WHITE RECT
0564 210000           LD   HL,0
0567 117100           LD   DE,113
056A CD6101           CALL MOVETO
056D 218002           LD   HL,640
0570 114100           LD   DE,65
0573 CD4802           CALL RFILL
0576 2604             LD   H,RED
0578 CD9C01           CALL SCCOL
057B 211901           LD   HL,281
057E 110000           LD   DE,0
0581 CD6101           CALL MOVETO
0584 215000           LD   HL,80
0587 111F01           LD   DE,287
058A CD4802           CALL RFILL
058D 210000           LD   HL,0
0590 117D00           LD   DE,125
0593 CD6101           CALL MOVETO
0596 217F02           LD    HL,639
0599 112800           LD    DE,40
059C CD4802           CALL RFILL
059F CDA505           CALL LDEL
05A2 C3AF05           JP   FAN
               ;
05A5 0607      LDEL   LD   B,7

05A7 C5        FL1    PUSH BC
05A8 CDD502           CALL TDEL
05AB C1               POP  BC
05AC 10F9             DJNZ FL1
05AE C9               RET
               ;
               ;
               ; END OF FLAG
               ;
05AF CD2601    FAN    CALL INIT
05B2 2652             LD   H,82
05B4 CDBA01           CALL SSTLE
05B7 210100           LD   HL,1
05BA 221601           LD   (INC),HL
05BD 3E01             LD   A,GREEN
05BF 321801           LD   (COLL),A

05C2 3A1801    FANL   LD   A,(COLL)
05C5 67               LD   H,A
05C6 CD9C01           CALL SCCOL
               ;
05C9 210000           LD   HL,0
05CC 221901           LD   (X1),HL
05CF 214001    FAN1   LD   HL,320
05D2 119000           LD   DE,144
05D5 CD6101           CALL MOVETO
05D8 2A1901           LD   HL,(X1)
05DB 110000           LD   DE,0
05DE CD0502           CALL LTO
```

```
05E1 ED4B1601          LD    BC,(INC)
05E5 09                ADD   HL,BC
05E6 221901            LD    (X1),HL
05E9 017E02            LD    BC,638
05EC B7                OR    A
05ED ED42              SBC   HL,BC
05EF FACF05            JP    M,FAN1


05F2 210000            LD    HL,0
05F5 221B01            LD    (Y1),HL
05F8 214001    FAN2    LD    HL,320
05FB 119000            LD    DE,144
05FE CD6101            CALL  MOVETO
0601 2A1B01            LD    HL,(Y1)
0604 117F02            LD    DE,639
0607 EB                EX    DE,HL
0608 CD0502            CALL  LTO
060B EB                EX    DE,HL
060C ED4B1601          LD    BC,(INC)
0610 09                ADD   HL,BC
0611 221B01            LD    (Y1),HL
0614 011F01            LD    BC,287
0617 B7                OR    A
0618 ED42              SBC   HL,BC
061A FAF805            JP    M,FAN2


061D 217F02            LD    HL,639
0620 221901            LD    (X1),HL
0623 214001    FAN3    LD    HL,320
0626 119000            LD    DE,144
0629 CD6101            CALL  MOVETO
062C 2A1901            LD    HL,(X1)
062F 111F01            LD    DE,287
0632 CD0502            CALL  LTO
0635 ED4B1601          LD    BC,(INC)
0639 B7                OR    A
063A ED42              SBC   HL,BC
063C 221901            LD    (X1),HL
063F 30E2              JR    NC,FAN3


0641 211F01            LD    HL,287
0644 221B01            LD    (Y1),HL
0647 214001    FAN4    LD    HL,320
064A 119000            LD    DE,144
064D CD6101            CALL  MOVETO
0650 2A1B01            LD    HL,(Y1)
0653 110000            LD    DE,0
0656 EB                EX    DE,HL
0657 CD0502            CALL  LTO
065A EB                EX    DE,HL
065B ED4B1601          LD    BC,(INC)
065F B7                OR    A
0660 ED42              SBC   HL,BC
0662 221B01            LD    (Y1),HL
0665 D24706            JP    NC,FAN4
```

```
                         ;
0668 CD1504                      CALL RAND
066B E607                        AND  7
066D FE00                        CP   0
066F 2001                        JR   NZ,FAN5
0671 3C                          INC  A
0672 6F          FAN5            LD   L,A
0673 2600                        LD   H,0
0675 221601                      LD   (INC),HL
0678 3A1801                      LD   A,(COLL)
067B 3C                          INC  A
067C FE08                        CP   8
067E 2005                        JR   NZ,FAN6
0680 FE10                        CP   16
0682 2001                        JR   NZ,FAN6
0684 3C                          INC  A
0685 321801      FAN6            LD   (COLL),A
0688 FE0A                        CP   10
068A C2C205                      JP   NZ,FANL
                         ;
068D C30703                      JP   MAIN
0690 C9                          RET
                         ; *** DATA ***
                         ;
0691 5A000028    DAT1            DEFB 90,0,0,40,-60,0,0,40,-30,0,0,-80,0,0
069F 1E000014    DAT2            DEFB 30,0,0,20,-30,0,0,-20,0,0
06A9 1E00003C    DAT3            DEFB 30,0,0,60,60,0,0,20,-90,0,0,-80,0,0
06B7 1E00003C    DAT4            DEFB 30,0,0,60,30,0,0,-60,30,0,0,80,-90,0,0,-80,0,0
06C9 5A000014    DAT5            DEFB 90,0,0,20,-30,0,0,60,-30,0,0,-60,-30,0,0,-20,0,0
06DB 5A000050    DAT6            DEFB 90,0,0,80,-90,0,0,-80,0,0
06E5 1E000028    DAT7            DEFB 30,0,0,40,-30,0,0,-40,0,0
```

```
5 REM BASIC PLUTO DEMONSTRATION PROGRAM
10 DAT = 161: STAT = 160
20 BLACK = 0: GREEN = 1: BLUE = 2: CYAN = 3: RED = 4: YELLOW = 5
30 MAGENTA = 6: WHITE = 7
35 DIM S(100)
36 DIM PA(460)
40 GOTO 2000
60 REM INITIALISE
70 REM
80 IF INP(STAT)<128 THEN 80
90 FOR WI=1 TO 4: OUT STAT,0: NEXT WI
100 IF INP(STAT)<128 THEN 100
110 OUT DAT,166
120 RETURN
130 REM
140 REM  MOVETO(X,Y)
150 REM
160 IF INP(STAT)<128 THEN 160
170 OUT DAT,151
180 IF INP(STAT)<128 THEN 180
190 OUT DAT,X AND 255: OUT DAT,INT(X/256)
200 OUT DAT,Y AND 255: OUT DAT,INT(Y/256)
210 RETURN
220 REM
230 REM MOVREL(X,Y)
240 REM
250 IF INP(STAT)<128 THEN 250
260 OUT DAT,152
270 HT=INT(X/256): IF HT<0 THEN HT=256+HT
280 KT=INT(Y/256): IF KT<0 THEN KT=256+KT
290 IF INP(STAT)<128 THEN 290
300 OUT DAT,X AND 255: OUT DAT,HT
310 OUT DAT,Y AND 255: OUT DAT,KT
320 RETURN
330 REM
340 REM LINREL(X,Y)
350 REM
360 IF INP(STAT)<128 THEN 360
370 OUT DAT,157
380 HT=INT(X/256): IF HT<0 THEN HT=256+HT
390 KT=INT(Y/256): IF KT<0 THEN KT=256+KT
400 IF INP(STAT)<128 THEN 400
410 OUT DAT,X AND 255: OUT DAT,HT
420 OUT DAT,Y AND 255: OUT DAT,KT
430 RETURN
440 REM
450 REM LINEREL SHORT (X,Y)
460 REM
470 IF INP(STAT)<128 THEN 470
480 OUT DAT,158
490 IF INP(STAT)<128 THEN 490
500 HT=X: IF HT<0 THEN HT=256+HT
510 KT=Y: IF KT<0 THEN KT=256+KT
520 OUT DAT,HT: OUT DAT,KT
530 RETURN
540 REM
550 REM LINETO(X,Y)
560 REM
570 IF INP(STAT)<128 THEN 570
580 OUT DAT,128
590 IF INP(STAT)<128 THEN 590
600 OUT DAT,X AND 255: OUT DAT,INT(X/256)
```

```
610 OUT DAT,Y AND 255: OUT DAT,INT(Y/256)
620 RETURN
630 REM
640 REM SETCCOL (C)
650 REM
660 IF INP(STAT)<128 THEN 660
670 OUT DAT,137
680 IF INP(STAT)<128 THEN 680
690 OUT DAT,C
700 RETURN
710 REM
720 REM SETFCOL (C)
730 REM
740 IF INP(STAT)<128 THEN 740
750 OUT DAT,136
760 IF INP(STAT)<128 THEN 760
770 OUT DAT,C
780 RETURN
790 REM
800 REM SETBCOL (C)
810 REM
820 IF INP(STAT)<128 THEN 820
830 OUT DAT,135
840 IF INP(STAT)<128 THEN 840
850 OUT DAT,C
860 RETURN
870 REM
880 REM SETSTYLE (S)
890 REM
900 IF INP(STAT)<128 THEN 900
910 OUT DAT,138
920 IF INP(STAT)<128 THEN 920
930 OUT DAT,S
940 RETURN
950 REM
960 REM RFILL (W,H)
970 REM
980 IF INP(STAT)<128 THEN 980
990 OUT DAT,129
1000 IF INP(STAT)<128 THEN 1000
1010 OUT DAT,W AND 255: OUT DAT,INT(W/256)
1020 OUT DAT,H AND 255: OUT DAT,INT(H/256)
1030 RETURN
1040 REM
1050 REM PFILL
1060 REM
1070 IF INP(STAT)<128 THEN 1070
1080 OUT DAT,130
1090 RETURN
1100 REM
1110 REM PRINT MESSAGE (M$)
1120 REM
1130 FOR WI=1 TO LEN(M$)
1140 IF INP(STAT)<128 THEN 1140
1150 OUT DAT,ASC(MID$(M$,WI,1))
1160 NEXT WI
1170 RETURN
1180 REM
1190 REM DRAW SHAPE
1200 REM
```

```
1210 WI=1
1220 X=S(WI): Y=S(WI+1):
1230 IF X=999 THEN RETURN
1240 GOSUB 450
1250 WI=WI+2: GOTO 1220
1260 RETURN
1270 REM
1280 REM ALLOCATE PARTITION (W,H,NSYMS)
1290 REM
1300 IF INP(STAT)<128 THEN 1300
1310 OUT DAT,163
1320 IF INP(STAT)<128 THEN 1320
1330 OUT DAT,(W AND 255): OUT DAT,INT(W/256)
1340 OUT DAT,(H AND 255): OUT DAT,INT(H/256)
1350 OUT DAT,NSYMS
1360 IF INP(STAT)<128 THEN 1360
1370 P=INP(DAT)
1380 RETURN
1390 REM
1400 REM LOAD SYMBOL
1410 REM
1420 IF INP(STAT)<128 THEN 1420
1430 OUT DAT,160
1440 IF INP(STAT)<128 THEN 1440
1450 OUT DAT,SNO
1460 FOR WI=0 TO W*H - 1
1465 IF INP(STAT)<128 THEN 1465
1470 OUT DAT,PA (WI)
1480 NEXT WI
1490 RETURN
1500 REM
1510 REM SET CSP(P)
1520 REM
1530 IF INP(STAT)<128 THEN 1530
1540 OUT DAT,164
1550 IF INP(STAT)<128 THEN 1550
1560 OUT DAT,P
1570 RETURN
1580 REM
1590 REM INQUIRE CSP (RET IN P)
1600 REM
1610 IF INP(STAT)<128 THEN 1610
1620 OUT DAT,146
1630 IF INP(STAT)<128 THEN 1630
1640 P = INP(DAT): PRINT"CSP=";P
1650 RETURN
1660 REM
2000 GOSUB 60: REM INIT
2010 C=GREEN:GOSUB 640:W=640:H=288:GOSUB 960
2020 X=20:Y=10:GOSUB 140
2030 C=BLACK:GOSUB 640:W=600:H=268:GOSUB 960
2040 GOTO 2060: OR USE THESE ...
2050 L=1000: M=1000: N=1000: I=.8: J=.8: GOTO 2080
2060 INPUT"VIEW COORDS X,Y,Z";L,M,N
2070 INPUT"INCREMENTS X,Y";I,J
2080 S=L*L+M*M: R=SQR(S): T=S+N*N: Q=SQR(T)
2090 FOR XW=-12 TO 12 STEP I: G=2
2100 FOR YW=-12 TO 12 STEP J: GOSUB 2360: GOSUB 2160: NEXT YW
2110 NEXT XW
2120 FOR YW=-12 TO 12 STEP I: G=2
2130 FOR XW=-12 TO 12 STEP J: GOSUB 2360: GOSUB 2160: NEXT XW
2140 NEXT YW
2150 GOTO 2380
```

```
2160 Q=T-XW*L-YW*M-Z*N
2170 XX=INT(T*(YW*L-XW*M)*8/(R*Q)+300)
2180 YY=110-INT(3*Q*(Z*S-N*(XW*L+YW*M))/(R*Q))
2190 IF XX>639 THEN XX=639
2200 IF XX<1 THEN XX=1
2210 IF YY>287 THEN YY=287
2220 IF YY<1 THEN YY=1
2230 X=XX: Y=YY
2240 IF G>2 THEN 2260
2250 GOSUB 140: GOTO 2330
2260 C=RED: IF Z>-5 THEN C=BLUE
2270 IF Z>0 THEN C=MAGENTA
2280 IF Z>5 THEN C=GREEN
2290 IF Z>10 THEN C=YELLOW
2300 IF Z>15 THEN C=CYAN
2310 IF Z>20 THEN C=WHITE
2320 GOSUB 640: GOSUB 550
2330 G=5
2340 RETURN
2350 REM
2360 Z=400/(10+XW*XW+YW*YW)-10: RETURN
2370 REM
2380 REM TEXT
2390 S=128: GOSUB 880
2400 X=420: Y=200: GOSUB 140 :REM ORIGIN
2410 REM A
2420 P=255: GOSUB 1510
2430 X=-WL*5: Y=HL+5: GOSUB 230
2440 WL=8: HL=10
2450 C=MAGENTA: GOSUB 720
2460 M$="WAIT ": GOSUB 1110
2470 X=-WL*5: Y=HL: GOSUB 230
2480 C=CYAN: GOSUB 720
2490 M$="for this ...": GOSUB 1110
2500 REM A TIMES BOLD
2510 DATA 0,0,0,0,0,1,1,1,0,0,0,0,0,0,0
2520 DATA 0,0,0,0,0,1,1,1,0,0,0,0,0,0,0
2530 DATA 0,0,0,0,1,0,1,1,1,0,0,0,0,0,0
2540 DATA 0,0,0,0,1,0,1,1,1,0,0,0,0,0,0
2550 DATA 0,0,0,1,0,0,0,1,1,1,0,0,0,0,0
2560 DATA 0,0,0,1,1,1,1,1,1,1,0,0,0,0,0
2570 DATA 0,0,1,0,0,0,0,0,1,1,1,0,0,0,0
2580 DATA 0,0,1,0,0,0,0,0,1,1,1,0,0,0,0
2590 DATA 0,1,1,0,0,0,0,0,0,1,1,1,0,0,0
2600 DATA 1,1,1,1,0,0,0,0,1,1,1,1,1,0,0
2610 REM B TIMES BOLD
2620 DATA 1,1,1,1,1,1,1,1,1,1,0,0,0,0,0
2630 DATA 0,0,1,1,1,0,0,0,0,1,1,0,0,0,0
2640 DATA 0,0,1,1,1,0,0,0,0,1,1,0,0,0,0
2650 DATA 0,0,1,1,1,0,0,0,0,1,1,0,0,0,0
2660 DATA 0,0,1,1,1,1,1,1,1,1,0,0,0,0,0
2670 DATA 0,0,1,1,1,0,0,0,0,1,1,0,0,0,0
2680 DATA 0,0,1,1,1,0,0,0,0,0,1,1,0,0,0
2690 DATA 0,0,1,1,1,0,0,0,0,0,1,1,0,0,0
2700 DATA 0,0,1,1,1,0,0,0,0,1,1,1,0,0,0
2710 DATA 1,1,1,1,1,1,1,1,1,1,1,0,0,0,0
2720 REM C TIMES BOLD
2730 DATA 0,0,0,1,1,1,1,1,1,1,0,0,0,0,0
2740 DATA 0,0,1,1,0,0,0,0,0,1,1,0,0,0,0
2750 DATA 0,1,1,0,0,0,0,0,0,0,0,0,0,0,0
2760 DATA 1,1,1,0,0,0,0,0,0,0,0,0,0,0,0
```

```
2770 DATA 1,1,1,0,0,0,0,0,0,0,0,0,0,0,0
2780 DATA 1,1,1,0,0,0,0,0,0,0,0,0,0,0,0
2790 DATA 1,1,1,0,0,0,0,0,0,0,0,0,0,0,0
2800 DATA 0,1,1,0,0,0,0,0,0,0,0,0,0,0,0
2810 DATA 0,0,1,1,0,0,0,0,0,1,1,0,0,0,0
2820 DATA 0,0,0,1,1,1,1,1,1,1,0,0,0,0,0
2830 REM D TIMES BOLD
2840 DATA 1,1,1,1,1,1,1,1,1,0,0,0,0,0,0
2850 DATA 0,0,1,1,1,0,0,0,1,1,0,0,0,0,0
2860 DATA 0,0,1,1,1,0,0,0,0,1,1,0,0,0,0
2870 DATA 0,0,1,1,1,0,0,0,0,1,1,1,0,0,0
2880 DATA 0,0,1,1,1,0,0,0,0,1,1,1,0,0,0
2890 DATA 0,0,1,1,1,0,0,0,0,1,1,1,0,0,0
2900 DATA 0,0,1,1,1,0,0,0,0,1,1,1,0,0,0
2910 DATA 0,0,1,1,1,0,0,0,0,1,1,0,0,0,0
2920 DATA 0,0,1,1,1,0,0,0,1,1,0,0,0,0,0
2930 DATA 1,1,1,1,1,1,1,1,1,0,0,0,0,0,0
2940 REM E TIMES BOLD
2950 DATA 1,1,1,1,1,1,1,1,1,1,1,0,0,0,0
2960 DATA 0,0,1,1,1,0,0,0,0,0,1,1,0,0,0
2970 DATA 0,0,1,1,1,0,0,0,0,0,0,0,0,0,0
2980 DATA 0,0,1,1,1,0,0,0,0,0,1,0,0,0,0
2990 DATA 0,0,1,1,1,1,1,1,1,1,1,0,0,0,0
3000 DATA 0,0,1,1,1,0,0,0,0,0,1,0,0,0,0
3010 DATA 0,0,1,1,1,0,0,0,0,0,0,0,0,0,0
3020 DATA 0,0,1,1,1,0,0,0,0,0,0,0,0,0,0
3030 DATA 0,0,1,1,1,0,0,0,0,0,1,1,0,0,0
3040 DATA 1,1,1,1,1,1,1,1,1,1,1,0,0,0,0
3050 C=BLUE: GOSUB 720
3060 W=15: H=10: NSYMS=5: GOSUB 1280
3070 GOSUB 1510: REM SET CSP
3080 X=-WL*5: Y=HL+5: GOSUB 230
3090 WL=W: HL=H
3100 FOR  L=0 TO 4
3110 FOR I = 0 TO W*H -1
3120 READ PA(I): NEXT I
3130 SNO=L: GOSUB 1400: REM LOAD SYM
3140 NEXT L
3150 FOR I = 0 TO 4
3160 IF INP(STAT)<128 THEN 3160
3170 OUT DAT,I
3180 NEXT I
3190 END
```