MPS/CO Computer/71-3/Rev.1
17 September, 1971

PDS1 DISPLAYS - PROGRAMMER'S GUIDE

B. Carpenter

CONTENTS

/je

INTRODUCTION

This document is intended as a manual for programmers wishing to use the PS or PSB IMLAC PDS1 displays. It supersedes previous documents on the subject.

Part 1 consists of a general description of the system and of how to use the displays by programming the IBM 1800.

Part 2 is for the use of those who wish to write PDS1 programs, thus extending the standard facilities to suit their personal requirements. Other documents relevant to Part 2 are the manual "PDS1 Assembly Language PDSA" and two IMLAC manuals, "User's Reference Manual" and "Programmer's Supplement".

Distribution

S. Battisti
H. van der Beken
G.P. Benincasa
L. Bertuzzi
J. Bosser
M. Bouthéon
L. Burnod
G. Cuisinier
A. Daneels
H. von Eicken, DD
~~P. Heymans~~
D. Lamotte
D. ~~Lord, DD~~
K.O.H. Pedersen
~~C. Piney, DD~~
C. Poinard
J.P. Potier
E. Ratcliff
W. Remmer
K. Schindl
C. Serre
G. Shering
U. Tallgren
D. Williams, DD

1. PART 1

1.1 General description of system

Two IMLAC PDS1 display mini-computers are installed in the MCR. The first has two display consoles (with keyboards) attached to one processor unit and the second has one display console and keyboard. Both have 4 K of 16 bit core store with the possibility of an extension to 8 K. The first, which has a light pen, is dedicated to

a) PS operation (first console)

b) injection (second console).

The second PDS1 is dedicated to the PSB. The PS and injection consoles will back each other up and the two PDS1's will back each other up.

The two processors are connected to the IBM 1800 by 16 bit parallel interfaces and STAR so that information may be transferred at a rate of 40 K words/second in blocks of approximately 250 words.

The software available to make use of the PDS1's consists of four main parts:

1) communications routines in the IBM

2) EXEC - a program resident in the PDS1

3) Assembler program for PDS1 Assembly language

4) programs and routines to use the disk file on which PDS1 coreloads are stored.

(Items 3) and 4) are discussed in Part 2 of this document).

The normal procedure is that a program in a partition of the IBM uses the communications routines to transmit data to, and to receive data from, a particular PDS1. The EXEC in the PDS1 handles certain types of transmission automatically, but if the standard facilities are insufficient the user programmer may add to them by writing a PDS1 program which will, in effect, extend the EXEC. This subject is covered in Part 2.

1.2 Display and keyboard possibilities

The screen of each display is 20 cm (horizontal) by 30 cm (vertical). The analogue XY signals are produced by hardware vector generators operating under software control. This software is in the form of a"display list" in the PDS1 core store, which is executed by the display processor and modified by the mini-computer processor.

The EXEC program modifies the display list according to information received from the IBM or from the keyboard.

For character display, the screen provides a matrix of 54 lines of 80 characters (character size approximately 2.5 mm, similar to line printer).

The character set is:

letters	:	A - Z (upper case only)
digits	:	0 - 9
symbols	:	+ - . , : [ ] * / = > < %
format characters:		space, new line, tab.

Lines 1 - 50 are available for users. Lines 51 - 54 are reserved for the system. A tab character causes an immediate skip to character position 43 on the current line.

For graphical display, a resolution of ½ mm on the screen can be achieved. EXEC provides either point or histogram plots of one or two graphs on up to five sets of axes simultaneously.

The maximum amount of information that can be displayed at one time is limited by the core available for the display list and by the time avaible to display each frame. Some data are given in section 1.5

to help in estimating the limits in a particular case, but experimentation will always be necessary.

The keyboard provides all the displayable characters plus some others which are not used. In addition there are a number of control keys, some of which have predefined meanings, and a key "CTRL" which may be used to modify a character for control purposes in a way analagous to the SHIFT key. Upper and lower case letters are treated alike by EXEC. There is a list of control key allocations in the Appendix.

1.3 PDS1 Resident EXEC program

1.3.1 General

The EXEC consists of three main parts: a monitor, an interrupt servicing routine and a set of subroutines.

The monitor is a short looping program which checks certain software flags and carries out actions requested by the IBM or the keyboard, where these are not carried out at interrupt level.

There is only one interrupt level and therefore only one interrupt servicing routine, which handles the following:

a) IBM transmissions to PDS1

b) 33 1/3 Hz display frame synchronization

c) keyboards and cursor movement.

There are also eight subroutines which are capable of modifying the display and of transmitting data to the IBM.

1.3.2 Extending EXEC

The normal mainline program in the PDS1 is the looping monitor. Where this does not provide a particular facility, the user may put his own mainline program in the PDS1. This program may use the EXEC subroutines and other features of EXEC to give a high degree of flexibility. The user will not need to learn the details of display generation, and the main benefit will be more effective use of the keyboard.

1.4 Routines in the IBM

1.4.1 General

Data is transmitted between the IBM and a PDS1 in the form of a file. The file is preceded by a 16 bit word of which the top 8 bits are a control function (CF) and the bottom 8 bits are a word count (WC). The useful length of a file is limited to slightly less than 255 words and its format is defined by the CF bits.

Interrupts are also exchanged between the computers. The only one of these which affects the user is called PRQ, i.e. "PDS1 Request". A typical procedure is as follows. An IBM coreload transmits an EBCDIC file to the PDS1, requesting the PDS1 to return the changes made by the operator. The PDS1 displays the text, and the operator types in changes using the keyboard and cursor. When he has finished, he presses the "TRANSMIT" key. EXEC then sends a PRQ interrupt to the IBM. The interrupt servicing subroutine in the IBM reads in the PRQ and then the CF/WC word. The latter is placed in a skeleton flag in the IBM. The IBM user coreload picks up the CF/WC and calls a routine to read in the file of changes made by the operator. Note that the PDS1, not the IBM, determines which and how many data are sent.

The PDS1 interrupts are on Level 5. Difficulties may occur if a program at Level 5 or higher priority attempts to transmit to or receive from a PDS1.

The skeleton flag mentioned above serves both as an interrupt indicator and as a means of passing the CF/WC word to the user. There is an extra skeleton flag associated with the PDS1's, called the queue indicator. When the PDS1 send a PRQ interrupt, the coreload associated with each non-zero bit in the queue indicator is queued, and the bit is reset. The interrupt indicator is set in any case. It is a user responsibility to set the bit required in the queue indicator. Because of restrictions in MPX it is only possible to have a total of 16 such queueable coreloads.

1.4.2 Transmission of data to PDS1*

(Check latest version of calling sequence in subroutine library manual.)

```
                 CALL          PDSTX
                 DC            LIST
                 DC            COMPL
                 |
                 |
                 |
                 |
                 |
                 |
                 |
                 |
                 |
       LIST      DC          /XYYZ        TRANSMISSION CONTROL WORD
                 BSS         5
                 DC          /QQ00+N      CF/WC
                 BSS         6
       BUFR      BSS         N            MAXIMUM 244
                 |
                 |
                 |
                 |
                 |
       COMPL     DC          *-*          COMPLETION CODE
```

Transmission control word

1st digit	:	PRU number
2nd + 3rd digits	:	button number
4th digit	:	0 = PS PDS1 1 = PSB PDS1

The first three digits are included primarily as a user identification. A dummy code (with PRU number 0) may be used when no program request unit is involved.

*Also see description of PDSAL in the subroutine library manual

The routines in the IBM keep a record of the busy status of each PDS1. If a display is created by a call with a particular transmission control word, subsequent calls will be refused unless they provide the same word. The busy status is cleared by PDLIB (section 1.4.4).

PDS1 control function

These 8 bits, supplied by the user, are transmitted to the PDS1 at the head of the file. See Appendix for control function definitions. Only those marked with an asterisk are recognised and processed automatically by EXEC in the PDS1. The others can be exploited by writing a PDS1 program.

Completion code

-1 during transmission

0 after successful transmission

>0 indicates an error - see subroutine library manual for details

Format of buffer when CF indicates EBCDIC

```
BUFR    DC     first line        } present only if delete
        DC     number of lines   } specified in CF

        DC     line              } position of first charac-
        DC     character pos.    } ter to be inserted

        DC     word count

        EBC    .TEXT.
```

In the text, newlines and tabs may be represented by the normal EBCDIC codes, or by $ and ' (apostrophe) respectively. Character position 81 does not correspond to position 1 on the next line - you must include newline or $ characters. It is more efficient to use tabs than spaces.

Care must be taken to use the correct card code for brackets. Model 26 punches give the wrong code. The & sign may be used for +.

(If only the delete is required, give a dummy line number of zero only.)

Double size characters

To obtain double size characters, insert the control character "ENTER SCALE 2" in the EBC statement. Following characters will be double size. At the end of the double size message, you must insert the control character "EXIT SCALE 2". You must insert a space after every character, and two new lines instead of one. (M E S S A G E $ $).

There are therefore 25 lines of 40 double-size characters. You may combine normal and double characters in one message if required, as long as at the end of the message the PDS1 is in normal size.

ENTER SCALE 2 - - - - - -EBCDID / 5E , punch 11, 8, 6
EXIT SCALE 2 - - - - - EBCDIC / 5F,, punch 11, 8, 7.

Format of buffer when CF indicates graphics

If "delete" is specified in the CF, all graphics previously on the screen will be deleted. Drawing operations automatically delete previous graphs with the same logical number.

```
BUFR    DC        n        LOGICAL NUMBER OF AXES (0-4)

                { 1        DRAW AXES
        DC      { 0        AXES ALREADY DRAWN
                { -1       DELETE AXES AND GRAPHS
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
        DC        ln       LINE AND CHARACTER POSITION       }
        DC        pn       OF ORIGIN                         }
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -}
        DC        -x       LIMITS OF  X                      }
        DC        +x                                         }
        DC        npx      LENGTH OF X-AXIS ON SCREEN        }  Only present
                           (CHARACTER POSITIONS)             }  if axes are
                                                             }  to be drawn
        DC        ndx      NUMBER OF SUBDIVISIONS OF         }
                           X-AXIS                            }
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -}
        DC        -y     }                                   }
        DC        +y     }                                   }
        DC        npy    } SAME FOR  Y                       }
        DC        ndy    }                                   }
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
        DC        ng       NUMBER OF GRAPHS (0, 1 or 2)
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
        DC        m       *LOGICAL NUMBER OF GRAPH (0-1) }
                { gc      *POINT PLOT                    }
        DC      { -gc     *HISTOGRAM PLOT                }  For each
                                                         }  graph
        DC        pc       NUMBER OF POINTS              }
        BSS       2*pc     LIST OF PAIRS OF  X  AND  Y   }
                           VALUES
```

Example 1

Draw a set of axes, logical number 3, for displaying Y values from -300 to +300 and X values from 0 to 100.

```
*   m = o   -- points plotted as "X"
    m = 1   -- points plotted as "."
   gc = 1  --- delete previous points
   gc = 2  --- no delete of previous points
```

```
BUFR    DC      3           LOGICAL NUMBER
        DC      1
        DC      10        } NEAR TOP OF SCREEN,
        DC      1         } EXTREME LEFT
        DC      0         }
        DC      100       } X  LIMITS
        DC      80          FULL WIDTH OF SCREEN
        DC      10          10 SUBDIVISIONS
        DC      -300      }
        DC      300       } Y  LIMITS
        DC      20          20 LINES HEIGHT
        DC      12          MARK EVERY 50 UNITS
        DC      0           DRAW NO GRAPHS
```

Example 2

Draw or refresh a graph of 14 points, in histogram form, on these axes.

```
BUFR    DC      3
        DC      0
        DC      1           DRAW ONE GRAPH
        DC      0           LOGICAL NUMBER OF GRAPH
        DC      -1
        DC      14
        BSS     28          DATA
```

Note

The PDS1 uses single-length multiply and divide routines, so in order to get good results in graph plotting it is important to use data suitably scaled.

Each character or line space on the screen corresponds to 32 least significant bits in the PDS1. The scaling factor stored in the PDS1 for the x and y directions of each set of axes is:

$$s = \frac{npx * 32}{\Delta x} \quad \text{or} \quad \frac{npy * 32}{\Delta y}$$

Here Δx is the range of x values as defined when the axes are drawn, and npx is the total length of the x axis in character positions.

Δx and Δy must be less than 2560. The values of s must be less than 64. They are stored in fixed-point form in 14 bits, i.e. to an accuracy of ± 0.004.

Values at x and y outside the range of the axes will be plotted, unless they would be off-screen.

1.4.3 Receiving data from PDS1

(Check latest version of calling sequence in subroutine library manual.)

```
          CALL      PDSRX
          DC        LIST
          DC        COMPL
           |
           |
           |
LIST      DC        /XYYZ          TRANSMISSION CONTROL WORD
          DC        N+1            WORD COUNT +1
          DC        *-*
BUFR      BSS       N              MAXIMUM 254
           |
           |
           |
COMPL     DC        *-*            COMPLETION CODE
```

Transmission control word and completion code

As for PDS TX.

Word count

This is the bottom 8 bits of the CF/WC word provided in the skeleton flag (section 1.4.1). For CF definitions, see Appendix. Only "changed characters" is provided automatically by EXEC. The others can be exploited by writing a PDS1 program.

Format of buffer for changed characters

Two words in the buffer correspond to each character typed in by the operator.

```
Word 0  :  top 8 bits   :  line number in decimal   (1-60)
        :  bottom 8 bits: character position in decimal  (1-80)

Word 1  :  bottom 8 bits: EBCDIC character.
```

If a character is changed more than once there will be more than one entry in the buffer.

The maximum number of changes that can be stored in the buffer is 127. If this many are found (i.e. if the word count is 254), the operator must be given a chance to continue by transmitting a short message to the PDS1 (e.g. CONTINUE) with "return changes" mode indicated in the CF.

1.4.4 Clearing busy status of PDS1

```
               CALL      PDLIB
               DC        TRCON
               I
               I
               I
      TRCON    DC        /XYYZ        TRANSMISSION CONTROL WORD
                                      (AS FOR PDSTX)
```

This clears the busy status for any PDS1 in use by the calling program. It must be called when the user has finished with a display; otherwise future users will be blocked.

1.4.5 Checking the abort status of PDS1

```
               CALL      PDSAB
               DC        TRCON
               DC        LABEL
               I
               I
               I
      TRCON    DC        /XYYZ        TRANSMISSION CONTROL WORD
                                      AS FOR PDSTX)
```

If the PDS1 specified has sent an abort request, the calling program is continued at address LABEL.

An abort request is sent by EXEC when the operator keys in CONTROL/A. It could also be sent by a user's PDS1 program.

1.5 Length and duration of display list

The maximum core available in the PDS1 for the display list is 1750 words, and the maximum time per frame is 30 000 μsec. If these limits are exceeded, the display will not be correct. The following data may be used to estimate the core usage and time for a particular case.

Fixed and system area	166 words	2430 μsec
Each character (visible, space, tab)	1 word	20 μsec (average)
Each set of axes (no labels	100 words (average)	500 μsec (average)
Single point	5 words	18 μsec
Histogram point	5 words (average)	10 μsec (average)

Example

Fixed area	166 words	2430 μsec
500 characters	500	10000
2 axis sets	200	500
2 x 50 points	500	1800
2 x 20 histogram pts.	200	400
TOTAL	1566 words	15130 μsec

1.6 Errors

Bad calls, etc., return a suitable completion code in the IBM. However, the files transmitted to the PDS1 are not checked in the IBM, so if they are faulty the calling program in the IBM will not be informed. The following error messages may appear on the PDS1 screen system area:

```
*BAD  CALL
*CORE FULL
* ?   FILE    (Unknown CF on file from IBM)
* ?   INPUT   (Unknown interrupt or undefined control key).
```

1.7 Timing note

Information is not displayed instantaneously after transmission by the IBM. If the programmer does not allow time for the PDS1 software to edit the display list before the next transmission, his level will be suspended by PDSTX or PDSRX until the PDS1 sets a completion bit in its status word.

The time taken to display data depends on the position on the screen, on the amount of information involved, and on the existing display. An average figure might be 100 msec but the range of variation is great.

1.8 Back-up arrangements and faults

The transmission routines automatically attempt to use the other PDS1 if the unit requested is broken. The skeleton flags are treated as if back-up had not occurred, so the calling program need take no account of back-up arrangements.

On the PS PDS1, the same display is seen on both screens, but only one keyboard is in use at a time. When the operator keys in CONTROL/B on either keyboard, EXEC changes to the one not currently selected. EXEC initially selects keyboard A (on the central console). A user's PDS1 program can also change the keyboard selection.

In the event of a fault destroying EXEC, the operator may force a PDS1 cold start by pressing once the FCS button on the PDS1 interface. An interrupt coreload in the IBM will attempt to reload the PDS1 from disc. If this operation succeeds, the screen should be cleared and the message *WAITING should appear on the system area. If it fails, the PDS1 interface may be switched to "off-line". The back-up PDS1 will then be used.

PART 2

2.1 General

As indicated in section 1.3, the EXEC does not fill the entire PDS1 core memory and it does not handle all possible types of file. In particular, its use of the keyboard is rather restricted. By writing a program to fill the remaining core of the PDS1, the user can extend the facilities of the system. There are three main advantages to this:

1) direct use of keyboard and display without using computing time in the IBM

2) user definition of file format, if required

3) user definition of method of interaction between operator and console (question-and-answer, interpretive language, check-list, light-pen, etc.).

The user need only learn a part of the PDS1 assembly language and will not need to program the display directly. He will also need to know a little about the organization of EXEC in the PDS1.

2.2 PDS1 Processor and Assembly Language

The PDS1 processor unit has two important registers:

1) the program counter
2) the accumulator.

The program counter contains the core address of the instruction being executed. The accumulator is used in the normal way for arithmetic calculations. The carry from Bit 0 of the accumulator complements a special bit, the link. In rotate operations, the link is transferred to Bit 0 and Bit 15 is transferred to the link.

Instructions fall into four main classes:

1) addressed instructions (arithmetic and jump)

2) micro-instructions (tests and manipulations of the accumulator)

3) IOT instructions (input/output)

4) display processor instructions.

Full descriptions of the instructions are given in the PDSA Assembly Language Manual and in the manufacturer's manuals. For simple applications, IOT instructions and display processor instructions may be ignored.

Note that all PDS1 instructions are single-length. The address field of an instruction contains an 11-bit absolute address; therefore the core store is divided into 2 K pages for addressing purposes. Locations outside the current 2 K page must be addressed indirectly. Since user programs and EXEC both reside in the same 2 K area, indirect addressing is rarely necessary.

The Assembly language which is available at CERN allows the programmer to use symbolic names as addresses, to define buffers with BSS statements, etc., but does not have more complex features such as macros. It uses the same card format as the IBM 1800 Assembler, but in general the mnemonics are different.

2.3 Getting a program into a PDS1

The PDSA Assembly Language Manual describes how to build a PDS1 core load (PCL) and store it in the disc area reserved for PCL's.

Assuming that EXEC is running correctly in the required PDS1, a PCL may be loaded from the IBM by use of the library subroutine PDSCL. (Check latest version of calling sequence in subroutine library manual.)

```
            CALL     PDSCL
            DC       NAME
            DC       LIST
            DC       COMP
            '
            '
            '
            BSS  E   0
NAME        DN       CLNAM       NAME IN PACKED FORM
            '
COMP        DC       *-*         COMPLETION CODE
            '
LIST        DC       /XYYZ       TRANSMISSION CONTROL WORD
            BSS      256
```

NAME

This is the PCL name as defined when it was built.

COMPLETION CODE

As for PDSTX, plus 16 { Name does not exist
Disc fault.

PDTEM can be used to check the completion code.

TRANSMISSION CONTROL WORD

As for PDSTX.

2.4 Facilities offered to the user by EXEC in the PDS1

2.4.1 General

As discussed in section 1.3, EXEC carries out input/output operations and display modifications. This section describes how a user program employs these EXEC facilities.

2.4.2 Skeleton flags

Symbolically, skeleton flags are addressed as SKF+n. (See section 2.5 for values of EXEC symbols). These are the uses of the skeleton flags:

SKF+1	Normally zero. When non-zero, a file has arrived from the IBM whose control function is defined but which is not automatically displayed by EXEC (see Appendix for CF definitions). SKF+1 contains the CF/WC word, exactly as supplied to PDSTX in the IBM. The user should reset SKF+1 to zero. If the file is EBCDIC or graphical, it has been loaded into locations starting at address BUFR. After processing the file, the user must set a completion bit for the IBM by executing IOT)742. (For binary files, see section 2.7).
SKF+2	Normally zero. When non-zero, a character has arrived from the keyboard currently selected. SKF+2 contains the EBCDIC value of the character (-1 for delete, $ for newline, apostrophe for tab.). User should reset SKF+2.
SKF+3	Keyboard selection. 0 = Keyboard A, -1 = Keyboard B.
SKF+4	Communication between interrupt level and monitor (system use only).
SKF+5 SKF+6	Cursor movement indicators (system use only).
SKF+7	Normally zero. When non-zero, transmit key has been pressed. The user should reset SKF+7 to zero.
SKF+8	Communication between parts of ISSR (system use only).
SKF+9	Communication between EXEC routines (system use only).
SKF+10	(Other name RCMAD). Pointer for "return changes" mode. (System use only).
SKF+11	(Other name CURSL). Current line number of cursor.
SKF+12	Current character position of cursor.

2.4.3 Escapes

A user program normally monitors input from the IBM and the keyboard by use of SKF+1 and SKF+2. However, an "escape" facility is provided so that the user can write part of his program to run on interrupt level.

When an undefined CF is detected on a file sent by the IBM, the data is read into core starting at address BUFR. Then, still at interrupt level, the interrupt servicing subroutine executes the instruction JMP I UFESC, Normally the location UFESC contains the address of coding to display an error message and return to the mainline. During program initialisation the user may execute these instructions:

```
LAW     USADR
DAC     UFESC
```

Here USADR is the address of coding to handle the unknown file as the user wishes. Then, when such a file arrives, the ISSR will execute an effective JMP USADR. After processing the file, the user must set a completion bit for the IBM by executing IOT)742.

The user program must leave the interrupt level by executing JMP BACK. It is absolutely forbidden to call any EXEC subroutine while on interrupt level.

There are two other escapes which work analogously. An unknown interrupt (e.g. light-pen) causes the ISSR to execute JMP I UIESC, and an unknown control key causes JMP I UCESC. In this case the accumulator contains the binary value supplied by the keyboard.

2.4.4 Leaving a program

Normally, the last logical statement in a program should be JMP EXIT. This causes EXEC to restart its monitor loop, after clearing SKF+1 to SKF+10. There is no transmission to the IBM. The escape addresses (section 2.4.3) are reset to their normal values. The display is not cleared.

If the operator presses CONTROL/A the program will be aborted. EXEC will send an abort request to the IBM and will then execute JMP EXIT.

If the user program executes JMP ROM, the PDS1 will be completely reloaded from the disc. This exit should never be used unless EXEC has been overwritten. It is exactly equivalent to pressing the FCS button on the PDS1 interface hardware (section 1.8).

2.4.5 Monitor subroutine

The normal mainline program in the PDS1, when no user PCL is running, consists of two instructions:

```
JMS     MONIT
JMP     .-1
```

Thus the machine executes subroutine MONIT repetitively.

If the user wishes to retain the automatic displaying facilities provided by MONIT (as described in Part I), his program should also call MONIT repetitively. In theory any period of time could be left between calls, but if possible it should not exceed 50 msec. MONIT changes the contents of the accumulator, and there is no completion code.

2.4.6 EBCDIC subroutines

```
JMS     INSCH
DC      N           LINE
DC      M           POSITION
```

This subroutine displays the character whose EBCDIC value is contained in the accumulator at character position M on line N of the screen.

It returns with zero in the accumulator if OK and -1 if there is an error. (An error message will be displayed the next time MONIT is called.) A newline character or $ is not allowed.

```
           JMS     INSST
           DC      N          LINE
           DC      M          POSITION
           DC      LIST
           |
           |
           |
LIST       DC      WC         WORD COUNT (GREATER THAN ZERO)
           EBC     .TEXT.
```

This subroutine displays the text, starting at the line and position given. See section 1.4.2 for punching conventions etc. Error code as for INSCH.

```
           JMS     DELLN
           DC      N          FIRST LINE
           DC      NUM        NUMBER OF LINES
```

This deletes NUM lines starting at line N. Error code as for INSCH.

2.4.7 Graphics subroutine

```
           JMS     DRGRA
           DC      LIST
           |
           |
           |
LIST       BSS     N
```

This carries out a graphics drawing operation according to the contents of LIST. The format of LIST is identical to that given in section 1.4.2. Error code as for INSCH.

```
           JMS     DELGR
```

This deletes all graphics. No error code. Contents of accumulator changed.

2.4.8 Error Monitor subroutine

```
           JMS     ERMON
           DC      MESS
           |
           |
MESS       DC      WC
           EBC     .ALARM.
```

If the accumulator contains zero, the message supplied is displayed in the next position on the system area at the bottom of the screen. The message may be up to 80 characters, not including a new line.

If the accumulator has certain non-zero values, then no parameters are supplied and fixed messages are displayed:

-1	BAD CALL
-2	CORE FULL
-3	? FILE
-4	? INPUT
-5	WAITING

There is no completion code and the accumulator is changed.

2.4.9 Transmission to IBM

```
              JMS      TRFIL
              DC       LIST
              DC       WC           WORD COUNT
              :
              :
LIST          BSS      WC
```

The CF bits for the transmission are supplied in the top half of the accumulator. Control functions are defined in the Appendix. The format of the list is indicated by the CF. Only "return changes" format is defined for system use.

TRFIL sends a PRQ interrupt to the IBM. Execution of TRFIL is not normally completed until the user's program in the IBM has called PDSRX in response to the interrupt.

(However, if the CF indicates one-word transmission, by having the two most significant bits equal to 1Ø, there are no parameters and no response by a user's program in the IBM is required.)

TRFIL gives no completion code and the accumulator is changed.

2.5 Core available and addresses

The 4 K of core in the PDS1 is divided as follows:

EXEC	:	1580 words (approx.)
User area	:	470 words (approx.)
Communications buffer	:	255 words
Display list + miscellaneous	:	1793 words

Following is a list of EXEC names which are included in the Permanent Symbol Table of the assembler. These names may be used by programs exactly as if they were defined by EQU statements. Where a value is given, it is fixed permanently; the others may change whenever EXEC is updated. (For this reason, all PCL's must be re-assembled and re-built whenever the EXEC is modified.)

Name	Value if fixed	Remarks
SKF	16	Base address for skeleton flags
UFESC		Undefined file escape address location
UCESC		Unrecognised control key escape address location
UIESC		Unknown interrupt escape address location
BACK		Return address after interrupt level processing via an escape.
EXIT		EXEC restart address
ROM	32	Starting address of Read Only Memory
MONIT INSCH INSST DELLN DRGRA DELGR ERMON TRFIL		Subroutine entry points
BUFR	2048	Address of first location in IBM communications buffer (length of buffer 255 words)
FSTAD		Address of first location available for a new program

The first statement in a user's program must be ORG FSTAD, which sets the origin address of his program to the value FSTAD. The address of the last location used must not exceed 2047 (3777 octal) and the system does not check this. The user may use the IBM communications buffer himself provided that his use does not conflict with use by MONIT. Display operations carried out by MONIT and "return changes" mode both use BUFR.

2.6 Light-pen

The EXEC provides no facilities for the light-pen except the escape UIESC (see section 2.4.3).

To initialize the light-pen execute:

```
IOT     )132     CLEAR     FLAG
LAW     3
IOT     )141     ENABLE    INTERRUPT
```

At interrupt level (each time the light-pen interrupts) execute:

```
IOT     )132     CLEAR     FLAG
IOT     )171     READ      X      POSITION
:
:
IOT     )172     READ      Y      POSITION
:
:
JMP     BACK
```

To exit, execute:

```
IOF              MASK      INTERRUPT
JMP     EXIT
```

The X and Y positions are encoded as described on page V-6-3 of the Imlac manual. After decoding, they are related to screen positions as follows:

```
    Line N :  Y = 3584 - 64N
Position M :  X =  736 + 32M
```

2.7 Miscellaneous details

Some of the auto-index registers are used by EXEC and are therefore not available for other programs. Users may employ the four with octal addresses)14,)15,)16,)17.

Note that octal locations)4010 to)4017 in BUFR are also auto-index registers.

If an IBM program calls PDSTX with a control function specifying "binary file", the format of the LIST for PDSTX is slightly modified:

```
LIST      DC       /XYYZ           TRANSMISSION CONTROL WORD
          BSS      5
CFWC      DC       /QQOO+N         CF + WC
CA        DC       ADR1            CORE ADDRESS
SA        DC       ADR2            STARTING ADDRESS
          BSS      4
BUFR      BSS      N               BINARY FILE
```

In this case, the binary information is loaded into the PDS1 core at address ADR1. If the CF specifies "load into core and enter" the binary file is treated as a program and entered at address ADR2. If the CF specifies "load and chain" the PDS1 stays at interrupt level and awaits another file, sent by another call to PDSTX. (This call to PDSTX must not send an interrupt to the PDS1. Bit 13 of the transmission control word is set to 1 to suppress the interrupt.) If the CF specifies "load and return to mainline", SKF+1 is set as described in section 2.4.2. (In this case the three words CFWC, CA, SA will be in locations CFA, CFA+1, CFA+2 in the PDS1; CFA EQU 5. The user does not need to execute IOT)742.)

More detailed information on any part of EXEC should be obtained from the listings and flowcharts.

APPENDIX

CONTROL FUNCTIONS FOR FILES FROM IBM TO PDS1 (8 BITS)

00000000	Cold start program
00xxxxxx	Undefined (free for users to define)
01xxxxxx	Alarm message
10	Binary file follows (for program loading):
10xxxx	Load into core, set SKF
11xxxx	Load into core, enter
0xxxxx	Load into core, chain
110	EBCDIC file
0xxxx	Load into buffer, set SKF
10000	*Display via EXEC (Hexadecimal /D000)
10100	*Display via EXEC, with initial delete (Hexadecimal /D400)
11000	*Display via EXEC, return changes (Hexadecimal /D800)
11100	*Display via EXEC, with initial delete, and return changes (Hexadecimal /DC00)
111	Graphical file
0xxxx	Load into buffer, set SKF
10000	*Display via EXEC (Hexadecimal /F000)
11000	*Display via EXEC, with initial delete (Hexadecimal /F800)

* Processed automatically by EXEC monitor

CONTROL FUNCTIONS FOR FILES FROM PDS1 TO IBM (8 BITS)

00000000	Not allowed
0xxxxxxx	Undefined (free for users to define)
10000000	Abort request
10100000	Queue request (no file involved)
11	Files:
000000	*Changed characters file (Hexadecimal /C000)
01xxxx	EBCDIC
10xxxx	STARC format
11xxxx	Undefined

* Provided automatically by EXEC monitor

Control key allocations

↑ ↓ → ←	Initiate cursor movement
CONT 5	Rapid cursor movement (normally single shot)
CONT 6	Stop cursor movement
CR or LF	New line
TAB	Skip to character position 43
DEL	*Delete. Converted to a "pseudo-EBCDIC". Value of -1
TRANSMIT	Send changes to IBM

* Not recognised by EXEC in "return changes" mode

Two-key allocations

↑A	Abort, restart EXEC monitor, send abort request to IBM
↑B	Change to other keyboard (PS PDS1 only)

This leaves 8 control keys and about 50 two-key combinations free for allocation by user programs in the PDS1.