TECHNICAL DESCRIPTION

OF DOS AND RTE

DRIVER FOR HP7210A

GRAPHIC PLOTTER

Part Number 17210-90004

TABLE OF CONTENTS

HP Computer Museum
www.hpmuseum.net

For research and education purposes only.

INTRODUCTION

Development and testing of the DOS* and RTE drivers for the HP7210A is now complete and the following BINARY tapes have been released.

DOS III complete HP7210A Plotter Driver (DVR10) Binary, 07210-16001 REV A

DOS III Minimum HP7210A Plotter Driver (DVR10) Binary, 07210-16002 REV A

RTE Complete HP7210A Plotter Driver (DVR10) Binary, 72008-60001 REV A.

RTE Minimum HP7210A Plotter Driver (DVR10) Binary, 72009-60001 REV A.

A description of the functional differences between the DOS and RTE drivers and the BCS driver is as follows:

DOS AND RTE DRIVER COMMANDS; Function Characteristics

FORTRAN, ALGOL

The FORTRAN and ALGOL usage by means of binary WRITE statements is identical for the BCS MINIMUM, the DOS MINIMUM, and the RTE MINIMUM drivers and is that usage described for the BCS MINIMUM driver in the HP1721Ø interface kit manual.

The FORTRAN and ALGOL usage by means of binary and formatted WRITE statements is identical for the BCS COMPLETE, the DOS COMPLETE, and the RTE COMPLETE drivers and is that usage described for the BCS COMPLETE driver in the HP1721Ø interface kit manual.

* These DOS III drivers are to be used in both the DOS-M and DOS-III operating systems.

ASSEMBLER - EXEC CALLS

All ASSEMBLER usage of the DOS MINIMUM, the DOS COMPLETE, the RTE MINIMUM and the RTE COMPLETE drivers is through EXEC calls and is distinct from the ASSEMBLER usage for the BCS MINIMUM and the BCS COMPLETE drivers. The EXEC call usage under DOS and RTE is as follows:

POSITION CALLS

ASSEMBLER EXEC - Call usage

```
EXT EXEC
--------
--------
--------
JSB EXEC
DEF * + 5                    Gives address of return point
DEF ICODE                    Gives address of request code
DEF ICNWD                    Gives address of control word
DEF IBUFR                    Gives address of 1st word of BUFFER
DEF IBFLN                    Gives address of BUFFER LENGTH
(Return Point)

ICODE DEC 2                  Defines request code
ICNWD OCT 00001UU            Defines control word
IBFLN DEC N (-2*N)           Defines BUFFER length
IBUFR DEC 0                  Defines 1st word of BUFFER (not used)
IPC   DEC 1 (0,-1)           Defines pen control word of BUFFER
IMC   DEC 1 (-1)             Defines mode control word of BUFFER
      BSS N-3                Provides space for point coordinate
```

FORTRAN EXEC Call usage (the use of WRITE statements as an alternative is strongly recommended).

```
DIMENSION IBUFR (N)
-------------------
-------------------
IBUFR (2) = IPC              Sets pen control value
IBUFR (3) = IMC              Sets mode control value
IBUFR (4) = IX(1)            Sets 1st X coordinate
IBUFR (5) = IY(1)            Sets 1st Y coordinate
IBUFR (6) = IX(2)            Defines optional additional coordinate
IBUFR (7) = IY(2)            Defines optional additional coordinate
-------------------
-------------------
```

```
----------
----------
IBUFR (2*(M+1)) = IX(M)          Defines optional additional coordinate value
IBUFR (2*(M+1)+1) = IY(M)        Defines optional additional coordinate value
IBUFLN = N (-2*N)                Sets buffer length
ICODE = 2                        Sets request code
ICNWD = 1UUB                     Sets control word
----------
----------
----------
```


```
Call Exec (ICODE, ICNWD, IBUFR, IBFLN)
```

ALGOL EXEC call usage (the use of WRITE statements as an alternative is strongly recommended).

```
HPAL,L, "PROGN," [Loader specs]
BEGIN
PROCEDURE EXECP (ECODE, ECNWD, EBUFR, EBFLN);
INTEGER ECODE, ECNWD, EBUFR, EBFLN; CODE;
INTEGER ICODE, ICNWD, IBFLN; IPC, IMC, I, N, M;
INTEGER ARRAY IBUFR (1:203); allocates buffer
INTEGER ARRAY IX (1:100); allocates X component storage
INTEGER ARRAY IY (1:100); allocates Y component storage

ICODE <-- 2;                     Defines request code
ICNWD <-- @1UU;                  Defines control word
IBUFR(2) <-- IPC;                Defines pen control buffer word
IBUFR(3) <-- IMC;                Defines mode control buffer word
IBUFR(4) <-- IX(1);              Defines 1st X component
IBUFR(5) <-- IY(1);              Defines 1st Y component
IBUFR(6) <-- IX(2);              Defines optional additional coordinate value
IBUFR(7) <-- IY(2);              Defines optional additional coordinate value
------------------
------------------
------------------
IBUFR (2*(M+1)) <-- IX(M);       Defines optional additional coordinate value
IBUFR (2*(M+1)+1) <-- IY(M);     Defines optional additional coordinate value
IBFLN <-- N;                     (-2*N) defines buffer length
-------------------------
-------------------------
EXEC (ICODE, ICNWD, IBUFR(1), IBFLN);
-------------------------
-------------------------
ENDS
HPAL, P, L, "EXECP", [Loader specs]
PROCEDURE EXECP (ECODE, ECNWD, EBUFR, EBFLN);
INTEGER ECODE, ECNWD, EBUFR, EBFLN;
CODE
END$
```

Position call usage is identical for MINIMUM and COMPLETE drivers.

ICODE must have the integer value 2 to specify a write. ICNWD must have bit 6=1 to specify binary transmission. For RTE bits 7 through 15 = 0. For DOS bits 7 through 11 = 0 bit 12 is the optional "WAIT" bit controling return of control from the driver and bits 13 through 15 = 0. IBFLN must give the buffer length for the points to be transmitted. It can have either a positive integer value giving the number of words or a negative integer value giving the number of characters. If m is the number of points to be transmitted, then:

IBFLN = 2 * m + 3 for words

IBFLN = -4 * m - 6 for characters

The buffer format is as follows:

Word 1	Not used	
Word 2	Pen control IPC	IPC > 0 down, IPC=0 points, IPC < 0 up
Word 3	Mode control IMC	IMC > 0 absolute IMC < 0 relative
Word 4	1st X component	Required
Word 5	1st Y component	Required
Word 6	Next X component	Optional
Word 7	Next Y component	Optional

The values in the buffer are unchanged on return from call.

LETTERING CALLS

ASSEMBLER EXEC Call usage

```
EXT EXEC
  ----
  ----
  ----
JSB EXEC
DEF * + 5                    Gives address of return point
DEF ICODE                    Gives address of request code
DEF ICNWD                    Gives address of control word
DEF IBUFR                    Gives address of 1st word of Buffer
DEF IBFLN                    Gives address of Buffer length
⟨Return Point⟩

                             a.  Binary size spec usage
ICODE DEC 2                  Defines request code
ICNWD OCT ØØØØUU             Defines control word
IBFLN DEC N (-2*N)           Defines buffer length

IBUFR DEC -1                 Value of 177777 defines binary size
IXX   DEC                    Value of size parameter
IXY   DEC                    Value of size parameter
IYX   DEC                    Value of size parameter
IYY   DEC                    Value of size parameter
      ASC N-5,-----          Text

                             b.  ASCII size spec usage
ICODE DEC 2                  Defines request code
ICNWD OCT ØØØØUU             Defines control word
IBFLN DEC N (-2*N)           Defines buffer length

IBUFR ASC 10, AAAAABBBBBCCCCCDDDDD Size spec.
      ASC N-10, -----        Text

                             c.  Old size spec usage

ICODE DEC 2                  Defines request code
ICNWD OCT ØØ1ØUU             Defines control word
IBFLN DEC N (-2*N)           Defines buffer length

IBUFR ------                 5 words containing the
 ------------                reformatting size spec
--------------               used by the previous
                             lettering call.

STEXT ASC N-5,-----          Text
```

Where STEXT is the symbol for location IBUFR + 4.

FORTRAN EXEC call usage (the use of WRITE statements as an alternative is strongly recommended).

```
      DIMENSION IBUFR(N), IAT (80), IAS (10)
FFFF  FORMAT (40A2)
      READ (LU,FFFF) (IAS(I), I=1,10) sames ASCII size spec
      READ (LU,FFFF) (IAT(I), I=1,40) sames text to be lettered
```

a. Binary size usage

```
      IBUFR(1) = -1                 Sets 1st buffer word for binary size spec
      IBUFR(2) = IXX                Sets size spec
      IBUFR(3) = IXY                Sets size spec
      IBUFR(4) = IYX                Sets size spec
      IBUFR(5) = IYY                Sets size spec
      DO TTTT I = 1,m
TTTT  IBUFR(I + 5)=ICA(I)           Places 2*m characters of text in buffer
      IBFLN = m + 5 (2*m-10)                sets buffer length
      ICODE = 2                     Sets request code
      ICNWD = LUP                   Sets control word

CALL EXEX (ICODE, ICNWD, IBUFR, IBFLN)
```

b. ASCII size usage

```
      DO SSSS I = 1,10              Sets size specification in buffer
SSSS  IBUFR(I) = IAS(I)
      DO TTTT I = 1,m               Sets 2m characters of text in
TTTT  IBUFR (I + 10)=IAT(I)         buffer

      IBFLN = m+10 (-2*m-20)        Sets buffer length
      ICODE = 2                     Sets request code
      ICNWD = LUP                   Sets control word
```

c. OLD size usage

```
      DO TTTT I = 1 m               Sets 2m characters of text in
TTTT  IBUFR(I+5)=IAT(I)             buffer

      IBFLN = m + 5 (-2*m-10)       Sets buffer length
      ICODE = 2                     Sets request code
      ICNWD = 0010000B              Sets control word
```

ALGOL EXEC call usage (the use of WRITE statements as an alternative is strongly recommended).

```
HPAL,L, "PROGN" [Loader specs]
BEGIN
PROCEDURE EXECC (ECODE, ECNWD, EBUFR, EBFLN);
INTEGER ECODE, ECNWD, EBUFR, EBFLN; CODE;
INTEGER ICODE, ICNWD, IBFLN, IXX, IXY, IYX, IYY, I, M, LUP
INTEGER ARRAY IBUFR (1:60);     allocate buffer
INTEGER ARRAY IAS (1:10);       allocate storage for ASCII size
INTEGER ARRAY IAT (1:40);       allocate storage for text
```

a. Binary size usage

```
IBUFR(1) <— -1;                 Sets 1st buffer word to indicate binary size
IBUFR(2) <—IXX;                 Sets size spec
IBUFR(3) <—IXY;                 Sets size spec
IBUFR(4) <—IYX;                 Sets size spec
IBUFR(5) <—IYY;                 Sets size spec

For I<—1 STEP 1 UNTIL M DO      Set text in buffer
IBUFR (I + 5) = IAT(I);         Set text in buffer

ICODE <—2;                      Set request code
ICNWD <—LUP;                    Set control word
IBFLN <—m + 5; (-2*m-10)                    Set buffer length

EXECC (ICODE, ICNWD, IBUFR(1), IBFLN);
```

b. ASCII size usage

```
For I<—1STEP1 UNTIL 10 DO  }    Set ASCII size spec
IBUFR(I) <—IAS(I);         }    Set ASCII size spec
For I<—1STEP1 UNTIL M DO   }    Set text in buffer
IBUFR(I+10) <—IAT(I);      }    Set text in buffer
ICODE <— 2;                     Set request code
ICNWD <—LUP;                    Set control word
IBFLN <—m + 10; (-2*m-20)       Set buffer length

EXECC (ICODE, ICNWD, IBUFR(1), IBFLN);
```

c. OLD size usage

```
For I<—1STEP1 UNTIL M DO        Set text in buffer
IBUFR(I + 5) <—IAT(I);          Set text in buffer
ICODE <—2;                      Set request code
ICNWD <—000010UU;               Set control word
IBUFLN <—m + 5 (-2*m-10)        Set buffer length

EXECC (ICODE, ICNWD, IBUFR(1), IBFLN);
```

```
END$
HPAL, P, L, "EXECC", <Loader specs>
PROCEDURE EXECC (ECODE, ECNWD, EBUFR, EBFLN);
INTEGER ECODE, ECNWD, EBUFR,EBFLN;
CODE
END$
```

Position call usage is only valid for complete drivers. In addition to the binary size and ASCII size usages which are essentially identical to the usages described for the BCS driver, there is an "OLD" size usage which is unique to the DOS and RTE drivers.

The reason for this is that upon return from a lettering call to the DOS or RTE driver the 1st five words of the buffer have been altered to contain the internal representation of the size spec. When "OLD" usage is specified the 1st five words of the buffer must contain an internal size specification generated by a previous lettering call and the text must start in word six of the buffer.

SUMMARY OF BUFFER FORMATS

a. Binary size

Word 1	-1	(177777_8)
Word 2	}	Size specs represented as signed binary integers
Word 3	}	Size specs represented as signed binary integers
Word 4	}	Size specs represented as signed binary integers
Word 5	}	Size specs represented as signed binary integers
Word 6		Text
---		---
---		---
---		---
---		---

SUMMARY OF BUFFER FORMATS CONTINUED

b. ASCII size

Word 1	20 ASCII character consisting of four each five character I5 format ASCII fields specifying four signed size specifications.
Word 2	
Word 3	
Word 4	
Word 5	
Word 6	
Word 7	
Word 8	
Word 9	
Word 10	
Word 11	Text
Word 12	Text
Word 13	Text
Word 14	Text
---	---
---	---
---	---
---	---

c. OLD size

Word 1	An internal size specification generated by or obtained from the buffer for a previous binary or ASCII lettering call
Word 2	
Word 3	
Word 4	
Word 5	

OTHER VALUES

ICODE = 2 (ØØØØØ2_8)	To specify a write request
ICNWD = LU (ØØØØUU_8)	the logical unit number of plotter for BINARY size and ASCII size lettering calls
$=\text{ØØ1ØUU}_8$	For "OLD" size lettering calls

The control word bits are assigned as follows

for RTE bits 15-1Ø must be set to Ø

Bit 9 = 0	For Binary and ASCII size

Bit 9 = 1	For "Old" size usage
Bits 8-7 = 0	
Bit 6	Must be set = 1 to specify a lettering call
Bits 5-0	Give plotter logical unit number

FOR DOS

Bits 15-14	Must = 0
Bit 13	If 1 specifies without unit If 0 specifies with wait
Bits 12-10	Must = 0
Bit 9	If 0 Binary or ASCII size If 1 "OLD" size
Bit 8-7	Must = 0
Bit 6	Must = 0 to specify lettering call
Bits 5-0	Give plotter logical unit number

IBFLN=

Total buffer length as a positive integer to specify a word count as a negative integer to specify a character count.	a. for BINARY size includes 5 words or 10 characters for size spec b. for ASCII size includes 10 words or 20 character for size specs c. for OLD size includes 5 words or 10 characters for size specs;

NOTE: To produce lettered text containing an odd number of characters, the character count option must be used.

As with the ECS driver, every line of lettering is followed by a simulated carriage return line feed not contained in the buffer. A buffer containing a size spec but no text will generate a carriage return and line feed and a buffer with a text ending in a back arrow (←) will not generate a terminal carriage return line feed.

STATUS CALLS

ASSEMBLER EXEC - Call Usage

STATUS CALLS CONTINUED

```
        EXT EXEC
        --------
        --------
        ---------
        JSB EXEC
        DEF *+5
        DEF ICODE                   Gives address of request code
        DEF ICNWD                   Gives address of control word
        DEF IRWD1                   Gives address for 1st word return
        DEF IRWD2                   Gives address for 2nd word return
       (Return Point)

        ICODE DEC 13                Defines request code
        ICNWD DEC LUP               Defines control word
        IRWD1 BSS 1                 1st status word returned here
        IRWD2 BSS 1                 2nd status word returned here
```

FORTRAN EXEC call usage

```
        ICODE = 13                  Defines request code
        ICNWD = LUP                 Defines control word

        CALL EXEC (ICODE, ICNWD, IRWD1, IRWD2)
```

ALGOL EXEC Call usage

```
        NPAL, L, "PROGN", (Loader specs)
        BEGIN
        PROCEDURE EXECS (ECODE, ECNWD, ERWD1, ERWD2);
        INTEGER ECODE, ECNWD, ERWD1, ERWD2; CODE;
        INTEGER ICODE, ICNWD, IRWD1, IRWD2, LUP;

        ICODE←13;                   Sets request code
        ICNWD← LUP;                 Sets control word

        EXEC (ICODE, ICNWD, IRWD1, IRWD2);
           ------
           ------
           ------
           ------
        END$
           ----
           ----
        NPAL, P, L, "EXECS", (Loader specs)
        PROCEDURE EXECS (ECODE, ECNWD, ERWD1, ERWD2);
        INTEGER ECODE, ECNWD, ERWD1, ERWD2;
        CODE
        END$
```

Status requests are processed by the operating system rather than the driver, but some aspects of the values returned are influenced by the driver.

SUMMARY OF VALUES RETURNED

The values returned by DOS and RTE are different and are as follows

DOS

(IRWD1) = Status Word
(IRWD2) = Transmission Log

The status word is EQT ENTRY 4 and bits 15-14 are availability

ØØ ⇒ available, Ø 1 ⇒ DOWN, and 1Ø ⇒ busy 11 ⇒ waiting for DMA and does not apply to plotter.

Bits 13-8 are TYPE CODE = ØØ1ØØØ for plotter

Bits 7-5 = 0 for plotter

Bit 4-0 have some significance as in BCS driver

The transmission log will contain a positive character on word count. Either a word or character count can occur for both position calls and lettering calls. The size specification is included for lettering calls, but simulated carriage returns and line feeds are not. The first unused word is included for position calls.

RTE

(IRWD1) = Word 5 of EQT ENTRY
(IRWD2) = Word 4 of EQT ENTRY

Word 5 of RTE's EQT is identical to the status word of DOS.

STATUS CALLS CONTINUED

Word 4 of RTE's EQT has the following contents

Bit 15	DMA usage	= 0 for plotter
Bit 14	BUFFERING	= 0 for no buffering = 1 to specify buffer
Bits 13-12	Not used	
Bit 11		= 1 if device timed out this will usually be result of a hardware failure if values provided by driver are unaltered.
Bits 10-9	Not used	
Bits 8-6	Unit number	Does not apply to plotter
Bits 15-0		Select code for this logical unit number.

GR:mh

January 9, 1973

CLEAR CALL (Used in DOS III only)

ASSEMBLER:

```
        EXT EXEC
          ---
          ---
          ---
        JSB EXEC
        DEF *+3
        DEF ICODE          Address of request code
        DEF ICNWD          Address of control word
          ---
          ---
          ---
ICODE   DEC 3              Specifies control request
ICNWD   DEC LU             Logical unit of plotter
```

FORTRAN:

```
        CALL EXEC(3,LU)
```

Where LU is the logical unit number of the plotter.

ALGOL:

```
        HPAL,L,"PROGC", <Loader Specs>
        BEGIN
        PROCEDURE EXEC3(ECODE,ECNWD);
        INTEGER ECODE,ECNWD;CODE;
        INTEGER ICODE,ICNWD,LU;

        ICODE ← -3;                Sets request code for control
        ICNWD ← LU;                Sets control word to logical unit

        EXEC3(ICODE,ICNWD);

        END$

        HPAL,P,L,"EXEC3", <Loader Specs>
        PROCEDURE EXEC3(ECODE, ECNWD);
        INTEGER ECODE,ECNWD;
        CODE
        END$
```