C 8

```
DDDDDDDDDDDD     RRRRRRRRRRRR        IIIIIIIII     VVV         VVV  EEEEEEEEEEEEEEE  RRRRRRRRRRRR
DDDDDDDDDDDD     RRRRRRRRRRRR        IIIIIIIII     VVV         VVV  EEEEEEEEEEEEEEE  RRRRRRRRRRRR
DDDDDDDDDDDD     RRRRRRRRRRRR        IIIIIIIII     VVV         VVV  EEEEEEEEEEEEEEE  RRRRRRRRRRRR
DDD         DDD  RRR         RRR        III        VVV         VVV  EEE              RRR         RRR
DDD         DDD  RRR         RRR        III        VVV         VVV  EEE              RRR         RRR
DDD         DDD  RRR         RRR        III        VVV         VVV  EEE              RRR         RRR
DDD         DDD  RRR         RRR        III        VVV         VVV  EEE              RRR         RRR
DDD         DDD  RRR         RRR        III        VVV         VVV  EEE              RRR         RRR
DDD         DDD  RRR         RRR        III        VVV         VVV  EEE              RRR         RRR
DDD         DDD  RRRRRRRRRRRR           III        VVV         VVV  EEEEEEEEEEEE     RRRRRRRRRRRR
DDD         DDD  RRRRRRRRRRRR           III        VVV         VVV  EEEEEEEEEEEE     RRRRRRRRRRRR
DDD         DDD  RRRRRRRRRRRR           III        VVV         VVV  EEEEEEEEEEEE     RRRRRRRRRRRR
DDD         DDD  RRR   RRR              III        VVV         VVV  EEE              RRR   RRR
DDD         DDD  RRR   RRR              III        VVV         VVV  EEE              RRR   RRR
DDD         DDD  RRR   RRR              III        VVV         VVV  EEE              RRR   RRR
DDD         DDD  RRR      RRR           III           VVV   VVV     EEE              RRR      RRR
DDD         DDD  RRR      RRR           III           VVV   VVV     EEE              RRR      RRR
DDD         DDD  RRR      RRR           III           VVV   VVV     EEE              RRR      RRR
DDDDDDDDDDDD     RRR         RRR     IIIIIIIII           VVV        EEEEEEEEEEEEEEE  RRR         RRR
DDDDDDDDDDDD     RRR         RRR     IIIIIIIII           VVV        EEEEEEEEEEEEEEE  RRR         RRR
DDDDDDDDDDDD     RRR         RRR     IIIIIIIII           VVV        EEEEEEEEEEEEEEE  RRR         RRR
```

L 10

```
**FILE**ID**DXPDRIVER
```

```
DDDDDDDD    XX      XX  PPPPPPPP    DDDDDDDD    RRRRRRRR      IIIIII    VV      VV  EEEEEEEEEE  RRRRRRRR
DDDDDDDD    XX      XX  PPPPPPPP    DDDDDDDD    RRRRRRRR      IIIIII    VV      VV  EEEEEEEEEE  RRRRRRRR
DD      DD  XX      XX  PP      PP  DD      DD  RR      RR      II      VV      VV  EE          RR      RR
DD      DD  XX      XX  PP      PP  DD      DD  RR      RR      II      VV      VV  EE          RR      RR
DD      DD    XX  XX    PP      PP  DD      DD  RR      RR      II      VV      VV  EE          RR      RR
DD      DD    XX  XX    PP      PP  DD      DD  RR      RR      II      VV      VV  EE          RR      RR
DD      DD      XX      PPPPPPPP    DD      DD  RRRRRRRR        II      VV      VV  EEEEEEEE    RRRRRRRR
DD      DD      XX      PPPPPPPP    DD      DD  RRRRRRRR        II      VV      VV  EEEEEEEE    RRRRRRRR
DD      DD    XX  XX    PP          DD      DD  RR  RR          II      VV      VV  EE          RR  RR
DD      DD    XX  XX    PP          DD      DD  RR  RR          II      VV      VV  EE          RR  RR
DD      DD  XX      XX  PP          DD      DD  RR    RR        II        VV  VV    EE          RR    RR        ....
DD      DD  XX      XX  PP          DD      DD  RR    RR        II        VV  VV    EE          RR    RR        ....
DDDDDDDD    XX      XX  PP          DDDDDDDD    RR      RR    IIIIII        VV      EEEEEEEEEE  RR      RR      ....
DDDDDDDD    XX      XX  PP          DDDDDDDD    RR      RR    IIIIII        VV      EEEEEEEEEE  RR      RR      ....


LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

Table of contents

```
0000           .TITLE  DXPDRIVER - VAX-11/780 RX01 CONSOLE DRIVER
0000           .IDENT  'V04-000'
0000
0000   ;
0000   ;****************************************************************************
0000   ;*                                                                          *
0000   ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000   ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000   ;*  ALL RIGHTS RESERVED.                                                    *
0000   ;*                                                                          *
0000   ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000   ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000   ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000   ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000   ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000   ;*  TRANSFERRED.                                                            *
0000   ;*                                                                          *
0000   ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000   ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000   ;*  CORPORATION.                                                            *
0000   ;*                                                                          *
0000   ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000   ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000   ;*                                                                          *
0000   ;*                                                                          *
0000   ;****************************************************************************
0000   ;
0000   ; C. A. MONIA 13-JUL-77
0000   ;
0000   ; MODIFIED BY:
0000   ;
0000   ;       V03-001 RAS0300         Ron Schaefer            19-Jun-1984
0000   ;               Add DEV$M_NNM characteristic to DECHAR2 so that these
0000   ;               devices will have the "node$" prefix.
0000   ;
0000   ;       V02-005 ACG0179         Andrew C. Goldstein,    23-Jul-1980  18:32
0000   ;               Fix ACP class code in DPT
0000   ;
0000   ;**
0000   ;
0000   ;
0000   ; STAR CONSOLE FLOPPY DISK DRIVER
0000   ;
0000   ; MACRO LIBRARY CALLS
0000   ;
0000
0000           $ADPDEF                         ;DEFINE ADP OFFSETS
0000           $CRBDEF                         ;DEFINE CRB OFFSETS
0000           $DDBDEF                         ;DEFINE DDB OFFSETS
0000           $DPTDEF                         ;DEFINE DPT OFFSETS
0000           $DYNDEF                         ;DEFINE DATA STRUCTURE TYPES
0000           $EMBDEF                         ;DEFINE EMB OFFSETS
0000           $IDBDEF                         ;DEFINE IDB OFFSETS
0000           $IODEF                          ;DEFINE I/O FUNCTION CODES
0000           $IRPDEF                         ;DEFINE IRP OFFSETS
0000           $UBADEF                         ;DEFINE UBA REGISTER OFFSETS
0000           $UCBDEF                         ;DEFINE UCB OFFSETS
```

```
          0000          $VECDEF                         ;DEFINE INTERRUPT DISPATCH VECTOR OFFSETS
          0000
          0000  ;
          0000  ; LOCAL SYMBOLS
          0000  ;
          0000  ; CONSOLE FLOPPY STATUS BIT DEFINITIONS
          0000  ;
          0000
          0000          _VIELD  DXP_RXDB,0,<-           ;RECEIVER DATA BUFFER FIELD DEFINITIONS
          0000          <CRC,,M>,-                      ;CRC ERROR
          0000          <PAR,,M>,-                      ;PARITY
          0000          <INI,,M>-                       ;INIT. COMPLETE
          0000  >                                       ;
          0000          _VIELD  DXP_RXDB,6,<-           ;
          0000          <DEL,,M>,-                      ;DELETED DATA MARK READ
          0000          <ERR,,M>-                       ;ERROR DETECTED
          0000  >                                       ;
          0000
          0000  ;
          0000  ; CONSOLE COMMAND FIELD DEFINITION
          0000  ;
          0000
00000800  0000  DXP_M_CMD = 8@8                         ;COMMAND BIT
          0000
          0000  ;
          0000  ; CONSOLE DATA BUFFER VALUES
          0000  ;
          0000
00000900  0000  DXP_RXDB_K_CMD = 9@8                    ;INITIATE FLOPPY FUNCTION
00000200  0000  DXP_RXDB_K_FCMP = 2@8                   ;FLOPPY FUNCTION COMPLETE
00000005  0000  DXP_RXDB_K_PRTC = 5                     ;PROTOCOL ERROR STATUS
          0000
          0000  ;
          0000  ; HARDWARE FUNCTION CODES
          0000  ;
          0000
00000800  0000  F_READSECTOR=8@8!0                      ;READ SECTOR
00000801  0000  F_WRITESECTOR=8@8!1                     ;WRITE SECTOR
00000802  0000  F_READSTATUS=8@8!2                      ;READ STATUS
00000803  0000  F_WRITEDELDATA=8@8!3                    ;WRITE DELETED DATA SECTOR
00000804  0000  F_CANCEL=8@8!4                          ;CANCEL FLOPPY FUNCTION
          0000
          0000  ;
          0000  ; LOCAL DATA
          0000  ;
          0000  ; DRIVER PROLOGUE TABLE
          0000  ;
          0000
          0000  DXP$DPT::                               ;
          0000          DPTAB   -                       ;DEFINE DRIVER PROLOGUE TABLE
          0000                  END=DXP_END,-           ;END OF DRIVER
          0000                  ADAPTER=NULL,-          ;ADAPTER TYPE
          0000                  FLAGS=DPT$M_SVP,-       ;SYSTEM PAGE TABLE ENTRY REQUIRED
          0000                  UCBSIZE=UCB$B_DX_SCTCNT+2,-   ; UCB SIZE
          0000                  NAME=DXDRIVER           ;DRIVER NAME
          0038          DPT_STORE INIT                  ;CONTROL BLOCK INIT VALUES
          0038          DPT_STORE DDB,DDB$L_ACPD,,,<^A\F11\> ;DEFAULT ACP NAME
```

```
003F          DPT_STORE DDB,DDB$L_ACPD+3,B,DDB$K_SLOW ;ACP CLASS
0043          DPT_STORE UCB,UCB$B_FIPL,B,8     ;FORK IPL
0047          DPT_STORE UCB,UCB$L_DEVCHAR,L,- ;DEVICE CHARACTERISTICS
0047                  <DEV$M_FOD-              ; FILES ORIENTED
0047                  !DEV$M_DIR-              ; DIRECTORY STRUCTURED
0047                  !DEV$M_AVL-              ; AVAILABLE
0047                  !DEV$M_SHR-              ; SHAREABLE
0047                  !DEV$M_IDV-              ; INPUT DEVICE
0047                  !DEV$M_ODV-              ; OUTPUT DEVICE
0047                  !DEV$M_RND>              ; RANDOM ACCESS
004E          DPT_STORE UCB,UCB$L_DEVCHAR2,L,-; DEVICE CHARACTERISTICS
004E                  <DEV$M_NNM>              ; PREFIX NAME WITH "node$"
0055          DPT_STORE UCB,UCB$B_DEVCLASS,B,DC$_DISK ;DEVICE CLASS
0059          DPT_STORE UCB,UCB$B_DEVTYPE,B,DT$_RX01   ;DEVICE TYPE
005D          DPT_STORE UCB,UCB$W_DEVBUFSIZ,W,512 ;DEFAULT BUFFER SIZE
0062          DPT_STORE UCB,UCB$B_SECTORS,B,26 ;NUMBER OF SECTORS PER TRACK
0066          DPT_STORE UCB,UCB$B_TRACKS,B,1   ;NUMBER OF TRACKS PER CYLINDER
006A          DPT_STORE UCB,UCB$W_CYLINDERS,W,77  ;NUMBER OF CYLINDERS
006F          DPT_STORE UCB,UCB$B_DIPL,B,20    ;DEVICE IPL
0073          DPT_STORE UCB,UCB$W_DEVSTS,W,UCB$M_NOCNVRT  ;NO LBN TO MEDIA ADDR. CONV.
0078          DPT_STORE UCB,UCB$B_ERTCNT,B,8  ;ERROR RETRY COUNT
007C          DPT_STORE UCB,UCB$B_ERTMAX,B,8  ;MAX ERROR RETRY COUNT
0080          DPT_STORE UCB,UCB$L_MAXBLOCK,L,<<76*26>/4>  ;MAX. NUMBER OF BLOCKS
0087          DPT_STORE REINIT                 ;CONTROL BLOCK RE-INIT VALUES
0087          DPT_STORE DDB,DDB$L_DDT,D,DXP$DDT ;DDT ADDRESS
008C          DPT_STORE END                    ;
0000
0000    ;
0000    ; DRIVER DISPATCH TABLE
0000    ;
0000
0000          DDTAB   DXP,-                   ;DRIVER DISPATCH TABLE
0000                  STARTIO,-               ;INITIATE I/O OPERATION
0000                  UNSOLNT,-               ;UNSOLICITED INTERRUPT
0000                  DX$FUNCTABLE,-          ;FUNCTION DECISION TABLE
0000                  UNSOLNT,-               ;DEASSIGN PROCESSING ROUTINE
0000                  REGDUMP,-               ;REGISTER DUMP ROUTINE
0000                  <<6>*4+<3>*4>,-         ;SIZE OF DIAGNOSTIC BUFFER
0000                  0,-                     ;SIZE OF ERROR BUFFER (0=NONE)
0000                  DX$UNITINIT             ;UNIT INITIALIZATION
```

```
0038            .SBTTL  START I/O OPERATION
0038
0038    ;+
0038    ; STARTIO - START I/O OPERATION ON DEVICE UNIT
0038    ;
0038    ; THIS ENTRY POINT IS ENTERED TO START AN I/O OPERATION ON THE CONSOLE
0038    ; FLOPPY. CONTROL ALTERNATES BETWEEN THE FLOPPY DRIVER AND THE FLOPPY
0038    ; UTILITY ROUTINES IN DXUTILITY THAT ARE SHARED BY ALL FLOPPY DEVICES.
0038    ; THE DRIVER PERFORMS ALL DEVICE-DEPENDANT PROCESSING WHILE THE UTILI-
0038    ; TY ROUTINES HANDLE THE INTERFACE BETWEEN THE DRIVE AND THE SYSTEM OR
0038    ; USER. THE PROTOCOL FOR EACH DATA TRANSFER IS AS FOLLOWS:
0038    ;
0038    ;       1. THE SYSTEM DETERMINES THE LEGALITY OF A FUNCTION AND ANY PRE-
0038    ;          PROCESSING TO BE PERFORMED BY SCANNING THE COMMON FUNCTION DE-
0038    ;          CISION TABLE.
0038    ;
0038    ;       2. A REQUEST IS DEQUEUED AND THE DRIVER IS ENTERED AT ITS START-I/O
0038    ;          ENTRY POINT.
0038    ;
0038    ;       3. A SUBROUTINE CALL TO COMMON CODE IS EXECUTED TO COMPUTE THE INI-
0038    ;          TIAL MEDIA ADDRESS AND SETUP THE I/O DATA BASE FOR THE TRANSFER.
0038    ;
0038    ;       4. A CO-ROUTINE CALL TO THE DRIVER IS EXECUTED TO POSITION THE MEDIA
0038    ;
0038    ;       5. A CO-ROUTINE CALL TO THE DRIVER IS PERFORMED TO TRANSFER ONE BYTE
0038    ;          OF DATA
0038    ;
0038    ;       6. A CO-ROUTINE CALL TO THE COMMON PROCESSOR IS PERFORMED TO UPDATE
0038    ;          THE ADDRESS AND BYTE COUNT AND CHECK FOR ENDPOINT CONDITIONS
0038    ;
0038    ; STEPS 4 - 6 ARE EXECUTED AT HARDWARE IPL LEVEL. A TRANSFER TO FORK LEVEL
0038    ; MUST BE EXECUTED WHENEVER:
0038    ;
0038    ;       . ONE SECTOR OF DATA HAS BEEN TRANSFERRED
0038    ;
0038    ;       . AN ERROR CONDITION IS DETECTED.
0038    ;
0038    ; ON OCCURANCE OF AN ERROR, THE DRIVER PERFORMS ANY HARDWARE DEPENDANT
0038    ; FUNCTIONS THEN TRANSFERS CONTROL TO THE COMMON EXCEPTION ENTRY POINT
0038    ; TO RESET MEDIA AND USER ADDRESSES AND REISSUE THE REQUEST.
0038    ;
0038    ; THE FLAGS DESCRIBED BELOW ARE USED TO SIGNAL EXCEPTION CONDITIONS:
0038    ;
0038    ;       FLAGS SET BY DRIVER:
0038    ;
0038    ;               WHEN EXCEPTION RETURN IS TAKEN:
0038    ;
0038    ;                       R3 LBS = RETRIABLE HARDWARE ERROR
0038    ;                       R3 LBC = FATAL HARDWARE ERROR
0038    ;
0038    ;                       R0 CONTAINS THE STATUS CODE REFLECTING THE TYPE OF ERROR
0038    ;                       DETECTED.
0038    ;
0038    ;       FLAGS SET BY UTILITY:
0038    ;
0038    ;               R3 LBC = TRANSFER OF ONE SECTOR COMPLETED
0038    ;
```

```
                         0038   ;               R3 LBS = PERFORM NORMAL DRIVER FUNCTION
                         0038   ;
                         0038   ;
                         0038   ; INPUTS:
                         0038   ;
                         0038   ;       R3 = ADDRESS OF I/O PACKET.
                         0038   ;       R5 = ADDRESS OF DEVICE UNIT CONTROL BLOCK
                         0038   ;
                         0038   ; OUTPUTS:
                         0038   ;
                         0038   ;       ********OUTPUTS********
                         0038   ;-
                         0038
                         0038           .ENABL  LSB
                         0038
                         0038   STARTIO:                                        ;
   00000000'EF      16   0038           JSB     DX$STARTIO                      ;CALL UTILITY ROUTINE TO SETUP PHYSICAL ADDR
                         003E   RESTART:                                        ;
  53   0800 8F      3C   003E           MOVZWL  #F_READSECTOR,R3                ;ASSUME READ PHYSICAL SECTOR
 03 68 A5     03    E1   0043           BBC     #UCB$V_DX_WRITE,UCB$W_DEVSTS(R5),20$ ;BRANCH IF READ
       53     01    80   0048           ADDB2   #<F_WRITESECTOR-F_READSECTOR>,R3 ;CONVERT FUNCTION CODE TO WRITE SEC
                         004B   20$:                                            ;
              63    10   004B           BSBB    DXPOUT                          ;SEND COMMAND TO FLOPPY
  53   00BC C5      9A   004D           MOVZBL  UCB$L_MEDIA(R5),R3              ;GET SECTOR NUMBER
              5C    10   0052           BSBB    DXPOUT                          ;OUTPUT SECTOR
  53   00BE C5      9A   0054           MOVZBL  UCB$L_MEDIA+2(R5),R3            ;GET CYLINDER NUMBER
              55    10   0059           BSBB    DXPOUT                          ;SEND CYLINDER NUMBER
 10 68 A5     03    E0   005B           BBS     #UCB$V_DX_WRITE,UCB$W_DEVSTS(R5),DXPWRITE ;BRANCH IF WRITE
              29    10   0060           BSBB    DXPINP                          ;WAIT FOR FLOPPY TO FINISH
                         0062   30$:                                            ;
              27    10   0062           BSBB    DXPINP                          ;WAIT FOR INPUT DATA
   00D0 D5    53    90   0064           MOVB    R3,@UCB$L_DX_BFPNT(R5)          ;STORE BYTE
              9E    16   0069           JSB     @(SP)+                          ;RETURN TO CALLER
           F4 53    E8   006B           BLBS    R3,30$                          ;IF LBS CONTINUE TRANSFER
              0E    11   006E           BRB     40$                             ;NO MORE DATA
                         0070
                         0070   ;+
                         0070   ; DXPWRITE - OUTPUT TO CONSOLE FLOPPY
                         0070   ;
                         0070   ; THIS ROUTINE IS ENTERED TO WRITE ONE BYTE OF DATA ON THE CONSOLE FLOPPY.
                         0070   ; IF AN ERROR OCCURS, A COROUTINE CALL IS MADE TO THE EXCEPTION ENTRY POINT
                         0070   ; SPECIFIED BY CALLER'S CALLER. THE STACK CONTAINS THE RETURN TO IN-LINE DRIVER
                         0070   ; CODE. IN THIS CASE, R3 INDICATES ERROR SEVERITY AS FOLLOWS:
                         0070   ;
                         0070   ;       R3 LBS   RETRIABLE ERROR
                         0070   ;       R3 LBC = FATAL ERROR
                         0070   ;
                         0070   ; THE ERROR CODE IS IN R0.
                         0070   ;
                         0070   ; IF NO ERROR OCCURS, A RETURN TO THE DRIVER IS EXECUTED.
                         0070   ;
                         0070   ; INPUTS:
                         0070   ;
                         0070   ;       R3 LBS = REQUEST TO WRITE ONE BYTE OF DATA
                         0070   ;       R5 = ADDRESS OF UCB
                         0070   ;       (SP) = RETURN ADDRESS
                         0070   ;
```

```
                        0070  ;       R3 LBC = SECTOR TRANSFER COMPLETE
                        0070  ;
                        0070  ; OUTPUTS:
                        0070  ;
                        0070  ;       A RETURN TO INLINE CODE IS EXECUTED FOR A SUCCESFUL TRANSFER.
                        0070  ;       IF SECTOR TRANSFER IS COMPLETE, A FORK IS EXECUTED BEFORE CALLING
                        0070  ;       THE CALLER.
                        0070  ;
                        0070  ;       A RETURN TO THE EXCEPTION ENTRY POINT IS TAKEN IF AN ERROR OCCURS.
                        0070  ;       IN THIS CASE, R3 INDICATES THE SEVERITY AND R0 CONTAINS THE ERROR
                        0070  ;       CODE.
                        0070  ;
                        0070  ;-
                        0070
                        0070  DXPWRITE:                                      ;
53     00D0 D5    9A    0070          MOVZBL  @UCB$L_DX_BFPNT(R5),R3     ;GET OUTPUT DATA
            39    10    0075          BSBB    DXPOUT                     ;SEND TO D' .ICE
            9E    16    0077          JSB     @(SP)+                     ;CALL THE CALLER
         F4 53    E8    0079          BLBS    R3,DXPWRITE                ;IF LBS WRITE ANOTHER BYTE
            0D    10    007C          BSBB    DXPINP                     ;WAIT FOR FLOPPY FUNCTION COMPLETE
                        007E  40$:                                           ;
     54     8E    D0    007E          MOVL    (SP)+,R4                   ;RETRIEVE RETURN ADDRESS
                        0C81          IOFORK                             ;DROP TO FORK LEVEL
            64    16    0087          JSB     (R4)                       ;CALL THE CALLER
            B3    11    0089          BRB     RESTART                    ;GO AGAIN
                        008B
                        008B          .DSABL  LSB
                        008B
                        008B  ;+
                        008B  ; DXPINP - WAIT FOR FLOPPY INPUT INTERRUPT
                        008B  ;
                        008B  ; THIS ROUTINE IS ENTERED VIA A BSB TO WAIT FOR AN INPUT INTERRUPT.
                        008B  ; IT ENTERS COMMON CODE TO EXECUTE A WAIT FOR INTERRUPT. ON RECEIPT
                        008B  ; OF THE INTERRUPT A RETURN TO THE CALLER IS EXECUTED AT ISR LEVEL.
                        008B  ;
                        008B  ; INPUTS:
                        008B  ;
                        008B  ;       (SP) = RETURN TO CALLER
                        008B  ;       4(SP) = RETURN TO CALLERS CALLER (DXUTILITY CO-ROUTINE)
                        008B  ;       8(SP) = RETURN TO EXECUTIVE
                        008B  ;
                        008B  ; OUTPUTS:
                        008B  ;
                        008B  ;       R3 = INPUT DATA
                        008B  ;       R5 = UCB ADDRESS
                        008B  ;
                        008B  ;       (SP) = RETURN TO CO-ROUTINE IN DXUTILITY
                        008B  ;
                        008B  ;-
                        008B
                        008B          .ENABL  LSB
                        008B
                        008B  DXPINP:                                        ;
                        008B          DSBINT  #31                        ;DISABLE ALL INTERRUPTS
25 64 A5    05    E0    0091          BBS     #UCB$V_POWER,UCB$W_STS(R5),5$ ;BRANCH IF POWER FAILURE
            1C    BA    0096          POPR    #^M<R2,R3,R4>              ;GET IPL IN R2, RETURNS IN R3, R4
            52    DD    0098          PUSHL   R2                         ;RESTORE IPL TO TOP OF STACK
```

```
                     009A             WFIKPCH 40$,#10                 ;WAIT FOR INTERRUPT
35 64 A5   07   E4   00A4             BBSC    #UCB$V_INTTYPE,UCB$W_STS(R5),25$ ;IF SET, RECEIVED INPUT INT.
   64 A5   03   A8   00A9             BISW    #<UCB$M_INT!UCB$M_TIM>,UCB$W_STS(R5) ;ENABLE INTERRUPTS AND TIMEOUTS
      00B5      31   00AD             BRW     UNSOLNT                         ;
                     00B0
                     00B0       ;+
                     00B0       ; DXPOUT - PERFORM OUTPUT TO THE CONSOLE FLOPPY DISK
                     00B0       ;
                     00B0       ; THIS ROUTINE IS ENTERED VIA A BSB TO SEND DATA TO THE CONSOLE FLOPPY.
                     00B0       . IT ENTERS THE COMMON CONSOLE INTERRUPT HANDLER TO TRANSFER THE DATA.
                     00B0       ;
                     00B0       ; INPUTS:
                     00B0       ;
                     00B0       ;         R3 = FLOPPY DATA (BIT 11 MUST BE SET FOR A FLOPPY CONSOLE COMMAND).
                     00B0       ;         R5 = ADDRESS OF UCB
                     00B0       ;         (SP) = RETURN TO CALLER
                     00B0       ;         4(SP) = RETURN TO CALLERS CALLER (DXUTILITY CO-ROUTINE)
                     00B0       ;         8(SP) = RETURN TO EXECUTIVE
                     00B0       ;
                     00B0       ; OUTPUTS:
                     00B0       ;
                     00B0       ;         R5 = UCB ADDRESS
                     00B0
                     00B0       ;         (SP) = RETURN TO CO-ROUTINE IN DXUTILITY
                     00B0       ;
                     00B0       ;-
                     00B0
                     00B0       DXPOUT:                                   ;
                     00B0             DSBINT  #31                     ;DISABLE ALL DEVICE INTERRUPTS
0A 64 A5   05   E1   00B6             BBC     #UCB$V_POWER,UCB$W_STS(R5),10$ ;BRANCH IF NO POWER FAILURE
                     00BB       5$:                                       ;
                     00BB             ENBINT                          ;ENABLE INTERRUPTS
           18   BA   00BE             POPR    #^M<R3,R4>              ;REMOVE RETURNS FROM STACK
           53   D4   00C0             CLRL    R3                      ;SET TO FLAG NONFATAL ERROR
         0088   31   00C2             BRW     50$                     ;
                     00C5       10$:                                      ;
  00000000'GF   16   00C5             JSB     G^CON$STARTIO           ;STARTUP THE DEVICE
                     00CB       20$:                                      ;
           1C   BA   00CB             POPR    #^M<R2,R3,R4>           ;GET IPL IN R2, RETURNS IN R3, R4
           52   DD   00CD             PUSHL   R2                      ;RESTORE IPL TO TOP OF STACK
                     00CF             WFIKPCH 40$,#10                 ;WAIT FOR INTERRUPT, KEEP CHANNEL
15 64 A5   07   E5   00D9             BBCC    #UCB$V_INTTYPE,UCB$W_STS(R5),30$ ;IF CLEAR, RECEIVED OUTPUT INTERRUP
                     00DE       25$:                                      ;
 00D4 C5   53   D0   00DE             MOVL    R3,UCB$L_DX_RXDB(R5)    ;SAVE CONTENTS OF RXDB
7E   53   04   08   EF   00E3             EXTZV   #8,#4,R3,-(SP)          ;EXTRACT COMMAND FIELD
      02   8E   D1   00E8             CMPL    (SP)+,#<DXP_RXDB_K_FCMP@-8>;TEST COMPLETION CODE
           06   19   00EB             BLSS    30$                     ;IF LSS PROCESS NEXT CHARACTER
           26   12   00ED             BNEQ    DXPERR                  ;IF NEQ, ERROR
           53   95   00EF             TSTB    R3                      ;TEST STATUS
           22   12   00F1             BNEQ    DXPERR                  ;IF NEQ, EXCEPTION CONDITION
                     00F3       30$:                                      ;
           54   DD   00F3             PUSHL   R4                      ;SET RETURN TO CALLER
        10 B5   17   00F5             JMP     @UCB$L_FR3(R5)          ;RETURN TO CALLER
                     00F8       40$:                                      ;
                     00F8             DSBINT  #31                     ;LOCKOUT ALL INTERRUPTS
06 64 A5   05   E0   00FE             BBS     #UCB$V_POWER,UCB$W_STS(R5),43$ ;DO NOT REINITIALIZE ON POWER FAIL
  00000000'GF   16   0103             JSB     G^CON$INITIAL           ;INITIALIZE DEVICE AND OUTPUT QUEUE
```

```
                                0109  43$:                                               ;
                                0109           ENBINT                                    ;ENABLE INTERRUPTS
                     53    D4   010C           CLRL     R3                               ;SET TO FLAG FATAL ERROR
          50   0000'8F     3C   010E           MOVZWL   #SS$_TIMEOUT,R0                  ;GET FINAL STATUS
                     3A    11   0113           BRB      60$                              ;
                                0115
                                0115  ;+
                                0115  ; DXPERR - HARDWARE DEPENDANT ERROR PROCESSING
                                0115  ;
                                0115  ; THIS ROUTINE IS ENTERED WHENEVER AN ERROR INDICATION IS RECEIVED FROM
                                0115  ; THE FLOPPY INTERFACE. IF THE PROBLEM WAS NOT CAUSED BY DEVICE TIMEOUT
                                0115  ; OR POWER FAIL THEN THE DEVICE REGISTERS ARE SAVED AND A FORK IS EXE-
                                0115  ; CUTED TO PERFORM ERROR ANALYSES.
                                0115  ;
                                0115  ; IF THE ERROR IS CAUSED BY HARDWARE, THE SEVERITY (FATAL OR NON-FATAL)
                                0115  ; AND STANDARD ERROR CODE ARE SETUP.
                                0115  ;
                                0115  ; INPUTS:
                                0115  ;
                                0115  ;        R3 = CONTENTS OF RXDB (IF ENTRY IS FROM CONSOLE INTERRUPT DISPATCHER)
                                0115  ;        R4 = CO-ROUTINE ENTRY POINT
                                0115  ;        R5 = ADDRESS OF UCB
                                0115  ;
                                0115  ; OUTPUTS:
                                0115  ;
                                0115  ;        R0 CONTAINS ERROR CODE
                                0115  ;        R3 LSB = 1, RETRIABLE ERROR
                                0115  ;        R3 LSB = 0, FATAL ERROR
                                0115  ;
                                0115  ;-
                                0115
                                0115  DXPERR:                                                ;
                                0115           IOFORK                                    ;FORK
          50   0000'8F     3C   011B           MOVZWL   #SS$_CTRLERR,R0                  ;ASSUME CONTROLLER ERROR
               51    53    D0   0120           MOVL     R3,R1                            ;COPY RXDB CONTENTS
                     53    D4   0123           CLRL     R3                               ;ASSUME ERROR IS FATAL
     52   51   04    08    EF   0125           EXTZV    #8,#4,R1,R2                      ;EXTRACT COMMAND FIELD
               02    52    91   012A           CMPB     R2,#<DXP_RXDB_K_FCMP@-8>  ;FUNCTION COMPLETE?
                     0C    13   012D           BEQL     45$                              ;IF EQL YES, CHECK ERRORS
               09    52    91   012F           CMPB     R2,#<DXP_RXDB_K_CMD@-8>  ;PROTOCOL ERROR?
                     1B    12   0132           BNEQ     60$                              ;IF NEQ GARBAGE IN SELECT FIELD
               05    51    91   0134           CMPB     R1,#DXP_RXDB_K_PRTC              ;CHECK DATA BYTE
                     14    13   0137           BEQL     50$                              ;IF EQL, PROTOCOL ERROR
                     14    11   0139           BRB      60$                              ;ELSE GARBAGE IN DATA BYTE
                                013B  45$:                                               ;
          50   0000'8F     3C   013B           MOVZWL   #SS$_PARITY,R0                   ;ASSUME DATA ERROR
                     51    95   0140           TSTB     R1                               ;TEST FOR DATA ERROR
                     09    19   0142           BLSS     50$                              ;IF LSS DATA ERROR
          50   0000'8F     3C   0144           MOVZWL   #SS$_FORMAT,R0                   ;ASSUME READ DELETED DATA
           02 51     06    E0   0149           BBS      #DXP_RXDB_V_DEL,R1,60$           ;IF BIT SET, FATAL FORMAT ERROR
                                014D  50$:                                               ;
                     53    D6   014D           INCL     R3                               ;NONFATAL ERROR
                                014F  60$:                                               ;
            00000000'EF    16   014F           JSB      DX$ERR                           ;CALL COMMON EXCEPTION CODE
                   FEE6    31   0155           BRW      RESTART                          ;RESTART TRANSFER
                                0158
                                0158           .DSABL   LSB
```

```
                             0158
                             0158  ;+
                             0158  ; REGDUMP - CONSOLE FLOPPY REGISTER DUMP ROUTINE
                             0158  ;
                             0158  ; THIS ROUTINE IS ENTERED TO COPY THE CONSOLE FLOPPY STATUS REGISTER
                             0158  ; CONTENTS (RXDB) TO THE SPECIFIED BUFFER. IT IS CALLED FROM THE DE-
                             0158  ; VICE ERROR LOGGING ROUTINE AND FROM THE DIAGNOSTIC BUFFER FILL ROU-
                             0158  ; TINE.
                             0158  ;
                             0158  ; INPUTS:
                             0158  ;
                             0158  ;       R0 = ADDRESS OF REGISTER SAVE BUFFER
                             0158  ;       R5 = ADDRESS OF UCB
                             0158  ;
                             0158  ; OUTPUTS:
                             0158  ;
                             0158  ;       THE COPY OF RXDB RECORDED IN THE UCB IS SAVED IN THE SPECIFIED BUFFER.
                             0158  ;
                             0158  ;-
                             0158
                             0158  REGDUMP:                                    ;
              80    02   D0  0158          MOVL    #2,(R0)+                    ;SET NUMBER OF DEVICE REGISTERS
        80    00D4 C5    D0  015B          MOVL    UCB$L_DX_RXDB(R5),(R0)+     ;COPY DEVICE REGISTER
        60    00BC C5    D0  0160          MOVL    UCB$L_MEDIA(R5),(R0)        ;COPY LAST DISK ADDRESS
                             0165  UNSOLNT:                                    ;
                         05  0165          RSB                                 ;
                             0166
                             0166  ;+
                             0166  ; DX$UNITINIT - UNIT INITIALIZATION
                             0166  ;
                             0166  ; THIS ROUTINE IS CALLED ON INITIAL DRIVER LOAD AND ON POWER RECOVERY
                             0166  ; TO INITIALIZE THE UNIT.  ON INITIAL DRIVER LOAD, IT ALLOCATES A
                             0166  ; 128 BYTE SECTOR BUFFER FROM NON-PAGED POOL AND LINKS IT ONTO THE UCB.
                             0166  ; IT THEN PUTS THE UCB ADDRESS INTO THE SLOT FOR UNIT 2 IN THE UCB LIST
                             0166  ; IN THE IDB.
                             0166  ; ON POWER RECOVERY IT SIMPLY RETURNS.
                             0166  ;
                             0166  ; INPUTS:
                             0166  ;
                             0166  ;       R5 = ADDRESS OF UCB
                             0166  ;
                             0166  ; OUTPUTS:
                             0166  ;
                             0166  ;       UCB$L_DX_BUF = ADDRESS OF SECTOR BUFFER
                             0166  ;-
                             0166
                             0166  DX$UNITINIT::
     44 64 A5     05     E0  0166          BBS     #UCB$V_POWER,UCB$W_STS(R5),30$   ; RETURN IF POWER RECOVERY
           00CC C5       D5  016B          TSTL    UCB$L_DX_BUF(R5)            ; IS THERE ALREADY A SECTOR BUFFER?
                3E       12  016F          BNEQ    30$                         ; YES, RETURN
                             0171
   54    00000000'GF     DE  0171          MOVAL   G^EXE$GL_NONPAGED,R4        ; PUT ADDRESS OF NON-PAGED POOL
                             0178                                              ; LIST HEAD IN R4
                64       DD  0178          PUSHL   (R4)                        ; SAVE IPL IN POOL LIST HEAD
   64    00000000'8F     DB  017A          MFPR    #PR$_IPL,(R4)               ; SET ALLOCATION IPL TO 31
                             0181
        51    8C 8F      9A  0181          MOVZBL  #140,R1                     ; SIZE OF BLOCK TO ALLOCATE
```

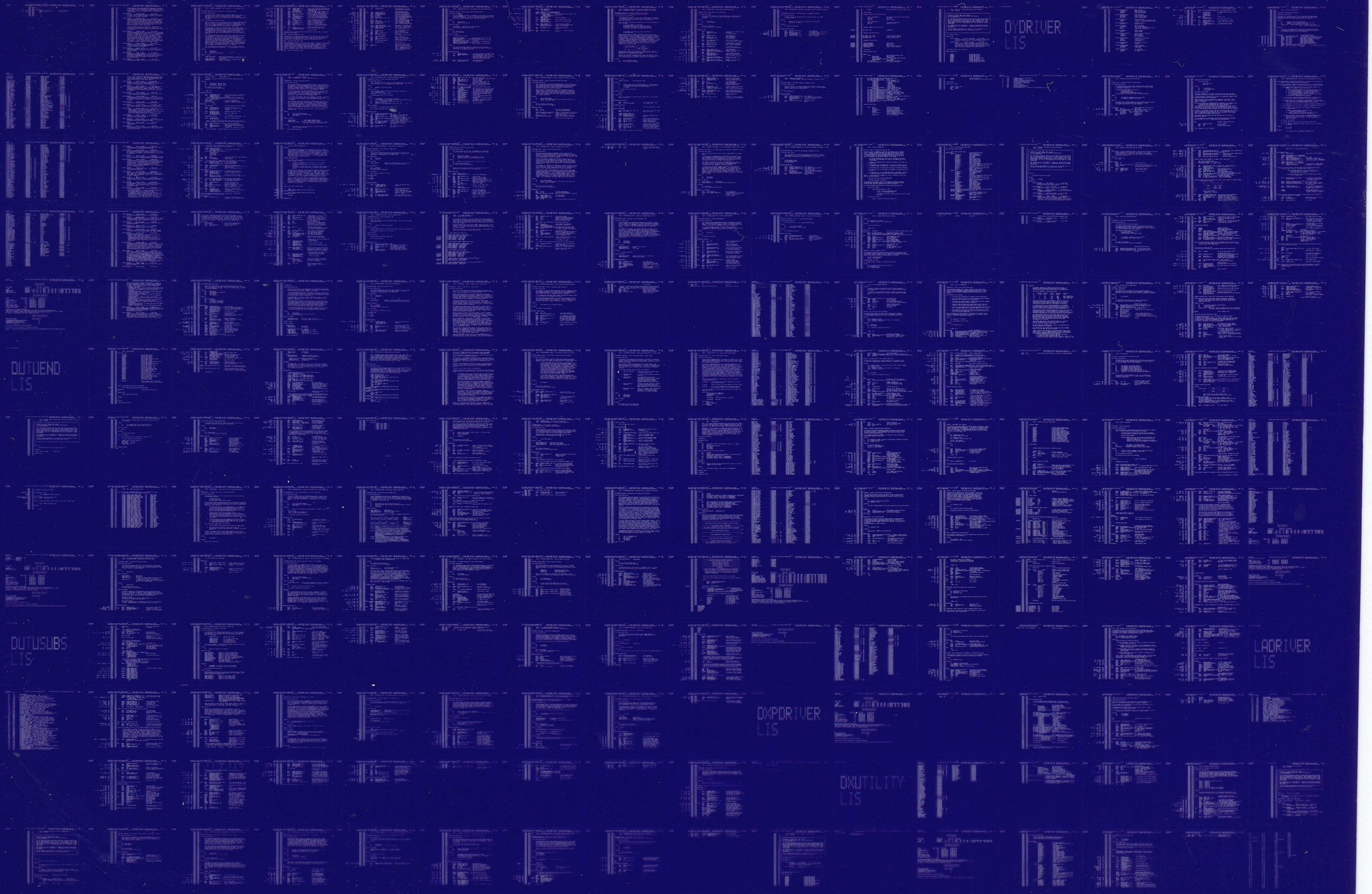
```
          00000000'GF     16  0185    498          JSB     G^EXE$ALONONPAGED          ; ALLOCATE MEMORY
                12 50     E9  018B    499          BLBC    R0,20$                     ; BR. IF FAILURE
          08 A2    51     B0  018E    500          MOVW    R1,8(R2)                   ; STORE SIZE OF BLOCK IN BLOCK
          0A A2    13     90  0192    501          MOVB    #DYN$C_BUFIO,10(R2)        ; STORE TYPE OF BLOCK IN BLOCK
   00CC C5    52   0C     C1  0196    502          ADDL3   #12,R2,UCB$L_DX_BUF(R5)    ; SAVE ADDRESS OF BLOCK
          64 A5    10     A8  019C    503          BISW    #UCB$M_ONLINE,UCB$W_STS(R5)  ; SET DEVICE ONLINE
                              01A0    504
                     64 8ED0  01A0    505 20$:     POPL    (R4)                       ; RESTORE IPL IN LISTHEAD
                              01A3    506
          50    24 A5     D0  01A3    507          MOVL    UCB$L_CRB(R5),R0           ; GET ADDRESS OF CRB
          51    2C A0     D0  01A7    508          MOVL    CRB$L_INTD+VEC$L_IDB(R0),R1  ; GET ADDRESS OF IDB
          20 A1    55     D0  01AB    509          MOVL    R5,IDB$L_UCBLST+8(R1)      ; STORE UCB ADDRESS IN SLOT FOR UNIT 2
                              01AF    510                                             ; (THIS IS BECAUSE CONSOLE FLOPPY CAN
                              01AF    511                                             ; INTERRUPT AS UNIT 2)
                              01AF    512
                          05  01AF    513 30$:     RSB
                              01B0    514
                              01B0    515          .END
```

```
$$$                     = 00000020 R       03    EXE$IOFORK                ********     X   04
$$OP                    = 00000002               FUNCTAB_LEN             = 00000000
AT$_NULL                  ********     X   03    F_CANCEL                = 00000804
BIT...                  = 00000008               F_READSECTOR            = 00000800
CON$INITIAL               ********     X   04    F_READSTATUS            = 00000802
CON$STARTIO               ********     X   04    F_WRITEDELDATA          = 00000803
CRB$L_INTD              = 00000024               F_WRITESECTOR           = 00000801
DC$_DISK                  ********     X   03    IDB$L_UCBLST            = 00000018
DDB$K_SLOW              = 00000003               IOC$MNTVER                ********     X   04
DDB$L_ACPD              = 00000010               IOC$RETURN                ********     X   04
DDB$L_DDT               = 0000000C               IOC$WFIKPCH               ********     X   04
DEV$M_AVL                 ********     X   03    PR$_IPL                   ********     X   04
DEV$M_DIR                 ********     X   03    REGDUMP                   00000158 R       04
DEV$M_FOD                 ********     X   03    RESTART                   0000003E R       04
DEV$M_IDV                 ********     X   03    SIZ...                  = 00000001
DEV$M_NNM                 ********     X   03    SS$_CTRLERR               ********     X   04
DEV$M_ODV                 ********     X   03    SS$_FORMAT                ********     X   04
DEV$M_RND                 ********     X   03    SS$_PARITY                ********     X   04
DEV$M_SHR                 ********     X   03    SS$_TIMEOUT               ********     X   04
DPT$C_LENGTH            = 00000038               STARTIO                   00000038 R       04
DPT$C_VERSION           = 00000004               UCB$B_DEVCLASS          = 00000040
DPT$INITAB                00000038 R       03    UCB$B_DEVTYPE           = 00000041
DPT$M_SVP               = 00000002               UCB$B_DIPL              = 0000005E
DPT$REINITAB              00000087 R       03    UCB$B_DX_SCTCNT         = 000000DA
DPT$TAB                   00000000 R       03    UCB$B_ERTCNT            = 00000080
DT$_RX01                  ********     X   03    UCB$B_ERTMAX            = 00000081
DX$ERR                    ********     X   04    UCB$B_FIPL              = 0000000B
DX$FUNCTABLE              ********     X   04    UCB$B_SECTORS           = 00000044
DX$STARTIO                ********     X   04    UCB$B_TRACKS            = 00000045
DX$UNITINIT               00000166 RG      04    UCB$L_CRB               = 00000024
DXP$DDT                   00000000 RG      04    UCB$L_DEVCHAR           = 00000038
DXP$DPT                   00000000 RG      01    UCB$L_DEVCHAR2          = 0000003C
DXPERR                    00000115 R       04    UCB$L_DX_BFPNT          = 000000D0
DXPINP                    0000008B R       04    UCB$L_DX_BUF            = 000000CC
DXPOUT                    00000080 R       04    UCB$L_DX_RXDB           = 000000D4
DXPWRITE                  00000070 R       04    UCB$L_FR3               = 00000010
DXP_END                   ********     X   03    UCB$L_MAXBLOCK          = 000000B0
DXP_M_CMD               = 00000800               UCB$L_MEDIA             = 000000BC
DXP_RXDB_K_CMD          = 00000900               UCB$M_INT               = 00000002
DXP_RXDB_K_FCMP         = 00000200               UCB$M_NOCNVRT           = 00000004
DXP_RXDB_K_PRTC         = 00000005               UCB$M_ONLINE            = 00000010
DXP_RXDB_M_CRC          = 00000001               UCB$M_TIM               = 00000001
DXP_RXDB_M_DEL          = 00000040               UCB$V_DX_WRITE          = 00000003
DXP_RXDB_M_ERR          = 00000080               UCB$V_INTTYPE           = 00000007
DXP_RXDB_M_INI          = 00000004               UCB$V_POWER             = 00000005
DXP_RXDB_M_PAR          = 00000002               UCB$W_CYLINDERS         = 00000046
DXP_RXDB_V_CRC          = 00000000               UCB$W_DEVBUFSIZ         = 00000042
DXP_RXDB_V_DEL          = 00000006               UCB$W_DEVSTS            = 00000068
DXP_RXDB_V_ERR          = 00000007               UCB$W_STS               = 00000064
DXP_RXDB_V_INI          = 00000002               UNSOLINT                  00000165 R       04
DXP_RXDB_V_PAR          = 00000001               VEC$L_IDB               = 00000008
DYN$C_BUFIO             = 00000013
DYN$C_DDB               = 00000006
DYN$C_DPT               = 0000001E
DYN$C_UCB               = 00000010
EXE$ALONONPAGED           ********     X   04
EXE$GL_NONPAGED           ********     X   04
```

```
                                            +----------------+
                                            ! Psect synopsis !
                                            +----------------+

PSECT name                       Allocation          PSECT No.  Attributes
----------                       ----------          ---------  ----------
.  ABS  .                        00000000  (    0.)  00 (  0.)  NOPIC    USR    CON    ABS    LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
. BLANK .                        00000000  (    0.)  01 (  1.)  NOPIC    USR    CON    REL    LCL NOSHR   EXE   RD     WRT NOVEC BYTE
$ABS$                            00000000  (    0.)  02 (  2.)  NOPIC    USR    CON    ABS    LCL NOSHR   EXE   RD     WRT NOVEC BYTE
$$$105_PROLOGUE                  0000008D  (  141.)  03 (  3.)  NOPIC    USR    CON    REL    LCL NOSHR   EXE   RD     WRT NOVEC BYTE
$$$115_DRIVER                    000001B0  (  432.)  04 (  4.)  NOPIC    USR    CON    REL    LCL NOSHR   EXE   RD     WRT NOVEC LONG

                                          +------------------------+
                                          ! Performance indicators !
                                          +------------------------+

Phase                     Page faults    CPU Time       Elapsed Time
-----                     -----------    --------       ------------
Initialization                     31    00:00:00.06    00:00:02.12
Command processing                120    00:00:00.39    00:00:03.27
Pass 1                            391    00:00:10.84    00:01:15.70
Symbol table sort                   0    00:00:01.55    00:00:12.68
Pass 2                            102    00:00:02.06    00:00:16.53
Symbol table output                14    00:00:00.07    00:00:00.07
Psect synopsis output               1    00:00:00.02    00:00:00.02
Cross-reference output              0    00:00:00.00    00:00:00.00
Assembler run totals              661    00:00:15.00    00:01:50.40

The working set limit was 1500 pages.
84630 bytes (166 pages) of virtual memory were used to buffer the intermediate code.
There were 80 pages of symbol table space allocated to hold 1480 non-local and 15 local symbols.
515 source lines were read in Pass 1, producing 17 object records in Pass 2.
34 pages of virtual memory were used to define 31 macros.

                                         +--------------------------+
                                         ! Macro library statistics !
                                         +--------------------------+

Macro library name                        Macros defined
------------------                        --------------
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                  23
_$255$DUA28:[SYSLIB]STARLET.MLB;2                6
TOTALS (all libraries)                          29

1723 GETS were required to define 29 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:DXPDRIVER/OBJ=OBJ$:DXPDRIVER MSRC$:DXPDRIVER/UPDATE=(ENH$:DXPDRIVER)+EXECML$/LIB
```

0111 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY


DYDRIVER
LIS

DUTUEND
LIS

DUTUSUBS
LIS

LADRIVER
LIS

DXPDRIVER
LIS

DXUTILITY
LIS