```
EEEEEEEEEEEEEEE  RRRRRRRRRRRR     FFFFFFFFFFFFFFF
EEEEEEEEEEEEEEE  RRRRRRRRRRRR     FFFFFFFFFFFFFFF
EEEEEEEEEEEEEEE  RRRRRRRRRRRR     FFFFFFFFFFFFFFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEEEEEEEEEEE     RRRRRRRRRRRR     FFFFFFFFFFFF
EEEEEEEEEEEE     RRRRRRRRRRRR     FFFFFFFFFFFF
EEEEEEEEEEEE     RRRRRRRRRRRR     FFFFFFFFFFFF
EEE              RRR   RRR        FFF
EEE              RRR   RRR        FFF
EEE              RRR   RRR        FFF
EEE              RRR      RRR     FFF
EEE              RRR      RRR     FFF
EEE              RRR      RRR     FFF
EEEEEEEEEEEEEEE  RRR         RRR  FFF
EEEEEEEEEEEEEEE  RRR         RRR  FFF
EEEEEEEEEEEEEEE  RRR         RRR  FFF
```

```
DDDDDDDD    RRRRRRRR    77777777    5555555555    000000
DDDDDDDD    RRRRRRRR    77777777    5555555555    000000
DD      DD  RR      RR          77  55          00      00
DD      DD  RR      RR          77  55          00      00
DD      DD  RR      RR          77  555555      00    0000
DD      DD  RR      RR          77  555555      00    0000
DD      DD  RRRRRRRR          77          55    00  00  00
DD      DD  RRRRRRRR          77          55    00  00  00
DD      DD  RR  RR          77              55  0000    00
DD      DD  RR  RR          77              55  0000    00
DD      DD  RR    RR      77        55      55  00      00      ....
DD      DD  RR    RR      77        55      55  00      00      ....
DDDDDDDD    RR      RR  77            555555      000000      ....
DDDDDDDD    RR      RR  77            555555      000000      ....


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
```

```
0001            Subroutine DR750 (lun)
0002    C
0003    C Version:      'V04-000'
0004    C
0005    C*********************************************************************
0006    C*                                                                   *
0007    C*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                          *
0008    C*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.           *
0009    C*  ALL RIGHTS RESERVED.                                             *
0010    C*                                                                   *
0011    C*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  *
0012    C*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
0013    C*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER  *
0014    C*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  *
0015    C*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY  *
0016    C*  TRANSFERRED.                                                        *
0017    C*                                                                   *
0018    C*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE  *
0019    C*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT  *
0020    C*  CORPORATION.                                                        *
0021    C*                                                                   *
0022    C*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS  *
0023    C*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.            *
0024    C*                                                                   *
0025    C*                                                                   *
0026    C*********************************************************************
0027    C
0028    C
0029    C
0030    C       Author: Sharon Reynolds                 Creation date:  21-July-1981
0031    C
0032    C       Functional Description:
0033    C
0034    C       This module decodes and outputs the error information for the
0035    C       DR750.
0036    C
0037    C       Modified by:
0038    C
0039    C       V03-004 SAR0219         Sharon A. Reynolds,     28-Mar-1984
0040    C               Changed the call to UCB$L_OWNUIC to ORB$L_OWNER.
0041    C
0042    C       V03-003 SAR0111         Sharon A. Reynolds,     23-Jun-1983
0043    C               Changed the carriage control in the 'format' statements
0044    C               for use with ERF.
0045    C
0046    C       V03-002 SAR0039         Sharon A. Reynolds,     8-Jun-1983
0047    C               Removed brief/cryptic support.
0048    C
0049    C       v03-001 BP0001          Brian Porter,           20-AUG-1982
0050    C               Minor edit.
0051    C**
0052    c--
0053
0054            Include 'SRC$:DR32COM.FOR /NOLIST'
0105            Include 'SRC$:DEVERR.FOR /NOLIST'
0206            Include 'SRC$:MSGHDR.FOR /NOLIST'
0265
```

```
      Byte            lun

      Integer*4       dr_dcr
      Integer*4       dr_uti
      Integer*4       dr_wcs_adr
      Integer*4       dr_wcs_dat
      Integer*4       dr_cmi_adr
      Integer*4       dr_cmi_bcr
      Integer*4       dr_di_bcr
      Integer*4       dr_dsl
      Integer*4       dr_sbr
      Integer*4       dr_gbr
      Integer*4       dr_cb(1:3)
      Integer*4       dr_bb(1:3)
      Integer*4       dr_gpr(0:22)

      Integer*4       silo_byte_cnt
      Integer*4       compressf
      Integer*4       compressc
      Integer*4       compress4
      Integer*4       field
      Integer*4       field1

      Real            float

      Parameter       dr_750 = 48

      Character*24    dr_dcr1(11:12)
      Character*17    dr_dcr2(14:20)
      Character*32    dr_dcr3(22:31)
      Character*24    bus_func(0:7)

C
C     Make the necessary information in the error log buffer available
C     to this module.
C
      Equivalence     (dr_dcr,EMB$L_DV_REGSAV(0))
      Equivalence     (dr_utl,EMB$L_DV_REGSAV(1))
      Equivalence     (dr_wcs_adr,EMB$L_DV_REGSAV(2))
      Equivalence     (dr_wcs_dat,EMB$L_DV_REGSAV(3))
      Equivalence     (dr_cmi_adr,EMB$L_DV_REGSAV(5))
      Equivalence     (dr_cmi_bcr,EMB$L_DV_REGSAV(6))
      Equivalence     (dr_di_bcr,EMB$L_DV_REGSAV(7))
      Equivalence     (dr_dsl,EMB$L_DV_REGSAV(8))
      Equivalence     (dr_sbr,EMB$L_DV_REGSAV(9))
      Equivalence     (dr_gbr,EMB$L_DV_REGSAV(10))
      Equivalence     (dr_cb,EMB$L_DV_REGSAV(11))
      Equivalence     (dr_bb,EMB$L_DV_REGSAV(14))
      Equivalence     (dr_gpr,EMB$L_DV_REGSAV(17))


      Data    dr_dcr1(11)     /'DATA INTERCONNECT STALL*'/
      Data    dr_dcr1(12)     /'JAM*'/


      Data    dr_dcr2(14)     /'NO CMI RESPONSE*'/
      Data    dr_dcr2(15)     /'RDS ERROR*'/
```

```
            Data    dr_dcr2(16)     /'CRD ERROR*'/
            Data    dr_dcr2(17)     /'HALT*'/
            Data    dr_dcr2(18)     /'DCR ABORT*'/
            Data    dr_dcr2(19)     /'PACKET INTERRUPT*'/
            Data    dr_dcr2(20)     /'INTERRUPT ENABLE*'/

            Data    dr_dcr3(22)     /'POWER UP*'/
            Data    dr_dcr3(23)     /'POWER DOWN*'/
            Data    dr_dcr3(24)     /'EXTERNAL ABORT*'/
            Data    dr_dcr3(25)     /'FORCE CI PARITY ERROR*'/
            Data    dr_dcr3(26)     /'FORCE DI PARITY ERROR*'/
            Data    dr_dcr3(27)     /'ENABLE DI PARITY ERROR ABORT*'/
            Data    dr_dcr3(28)     /'LOCAL PARITY ERROR*'/
            Data    dr_dcr3(29)     /'CI PARITY ERROR*'/
            Data    dr_dcr3(30)     /'DI PARITY ERROR*'/
            Data    dr_dcr3(31)     /'PARITY ERROR*'/

            Data    bus_func(0)     /'READ*'/
            Data    bus_func(1)     /'READ LOCK*'/
            Data    bus_func(2)     /'ILLEGAL FUNCTION*'/
            Data    bus_func(3)     /'ILLEGAL FUNCTION*'/
            Data    bus_func(4)     /'WRITE*'/
            Data    bus_func(5)     /'WRITE UNLOCK*'/
            Data    bus_func(6)     /'WRITE VECTOR*'/
            Data    bus_func(7)     /'ILLEGAL FUNCTION*'/


C
C       Force a top of form to occur and construct the report header.
C
            Call FRCTOF (lun)

            Call DHEAD1 (lun,'DR32')

C
C       Decode and output the DR750 Control Register.
C
            Call LINCHK (lun,2)
            Write (lun,20) dr_dcr
20          Format (/' ',T8,'DR DCR',T24,Z8.8)

            Field = LIB$EXTZV (0,8,dr_dcr)

            Call LINCHK (lun,1)
            If (field .ne. dr_750) then

            Write (lun,25)
25          Format (' ',T40,'ADAPTER NOT DR750')
            Else

            Write (lun,30)
30          Format (' ',T40,'ADAPTER IS DR750')
            Endif

            Call OUTPUT (lun,dr_dcr,dr_dcr1,11,11,12,'0')

            Call OUTPUT (lun,dr_dcr,dr_dcr2,14,14,20,'0')
```

```
            Call OUTPUT (lun,dr_dcr,dr_dcr3,22,22,31,'0')

C
C           Decode and output the DR750 Utility Register.
C
            Call LINCHK (lun,1)
            Write (lun,40) dr_utl
40          Format (' ',T8,'DR UTL',T24,Z8.8)

            Field = LIB$EXTZV (0,8,dr_utl)

c
c           See DR780 listing for explanation of speed calculation.
c

            If (field .le. 'FB'X) then

            Field = JIDIM (256,field)

            Float = FLOATJ(field)

            float = 40/float

            float = (1024*1024) / ((float*1024*1024)/4)

            Call LINCHK (lun,2)
            Write (lun,45) float
45          Format (' ',T40,'PERIOD = ',
            1 F<COMPRESSF (float,1)>.1,'. MICRO/SEC')

            Float = FLOATJ(field)

            float = (40/float)*1024

            Write (lun,50) float
50          Format (' ',T40,'TRANSFER = ',
            1 F<COMPRESSF (float,2)>.2,'. KB/SEC')

            Endif

            Field = LIB$EXTZV (11,1,dr_utl)

            Call LINCHK (lun,1)
            If (field .eq. 1) then

            Write (lun,60)
60          Format (' ',T40,'MICRO-CODE LOADED AND VALID')
            Endif

            Field = LIB$EXTZV (8,3,dr_utl)
            Field1 = LIB$EXTZV (12,4,dr_utl)

            Call LIB$INSV (field,0,3,silo_byte_cnt)
            Call LIB$INSV (field1,3,4,silo_byte_cnt)

            Call LINCHK (lun,2)
```

```
            Write (lun,70) silo_byte_cnt
70          Format (' ',T40,'SILO BYTE COUNT = ',
            1 i<compress4 (silo_byte_cnt)>,'.')


C
C           Output the DR750 WCS Registers.
C
            Call LINCHK (lun,2)
            Write (lun,80) dr_wcs_adr
80          Format (' ',T8,'DR WCS',T24,Z8.8,/,
            1 T40,'WCS ADDRESS REGISTER')

            Call LINCHK (lun,2)
            Write (lun,85) dr_wcs_dat
85          Format (' ',T8,'DR WCS',T24,Z8.8,/,
            1 T40,'WCS DATA REGISTER')

C
C           Decode and output the DR750 CMI Address Register.
C
            Call LINCHK (lun,1)
            Write (lun,100) dr_cmi_adr
100         Format (' ',T8,'DR BA',T24,Z8.8)

C
C           This is the CMI address
C
            Field = LIB$EXTZV (0,24,dr_cmi_adr)

C
C           Clear bits 0:1 of the CMI address
C
            Call LIB$INSV (0,0,2,field)

            Call LINCHK (lun,1)
            Write (lun,105) field
105         Format (' ',T40,'LONGWORD ADDRESS (HEX) = ',
            1 Z6)

C
C           This is the CMI bus function.
C
            Field = LIB$EXTZV (25,3,dr_cmi_adr)

            Call LINCHK (lun,1)
            Write (lun,110) bus_func(field)
110         Format (' ',T40,'CMI BUS FUNCTION = ',
            1 A<COMPRESSC (bus_func(field))> )

C
C           This is the Byte mask (the dr750 will always read 32 bits (longwords)
C           but can mask which bytes to write).
C
            Field = LIB$EXTZV (28,4,dr_cmi_adr)
```

```
C
C       Decode and output the DR750 CMI Byte Count Register and the Data
C       Interconnect Byte Count Register.
C
        Call LINCHK (lun,2)
        Write (lun,150) dr_cmi_bcr
150     Format (' ',T8,'DR_BC',T24,Z8.8,/,
        1 T40,'CMI BYTE COUNT')

        Call LINCHK (lun,2)
        Write (lun,160) dr_di_bcr
160     Format (' ',T8,'DR_BC',T24,Z8.8,/,
        1 T40,'DATA INTERCONNECT BYTE COUNT')

C
C        Decode and output the DR750 Status Longword
C
        Call LINCHK (lun,1)
        Write (lun,170) dr_dsl
170     Format (' ',T8,'DR_DSL',T24,Z8.8)

        Call OUTPUT (lun,dr_dsl,V1dr_sl,0,0,14,'0')

        Field = LIB$EXTZV (16,8,dr_dsl)

        If (field .ne. 0) then

        Call LINCHK (lun,3)
        Write (lun,175) ('*', I = 0,32)
175     Format (' ',T40,33A1,/,T47,'FAR-END STATUS BYTE',/)

        Do 185, I = 0,4

        Field1 =  LIB$EXTZV (I,1,field)

        If (field1 .ne. 0) then

        Call LINCHK (lun,1)
        Write (lun,180) I
180     Format (' ',T40,'FAR-END DEVICE STATUS BIT ',I1,'. SET')
        Endif

185     Continue

        Call OUTPUT (lun,dr_dsl,V2dr_sl,21,21,23,'0')

        Call LINCHK (lun,1)
        Write (lun,190) ('*', I = 0,32)
190     Format (' ',T40,33A1)
        Endif

C
C       Output the DR750 System Base Register
C
        Call LINCHK (lun,2)
        Write (lun,200) dr_sbr
```

```
200     Format (' ',T8,'DR SBR',T24,Z8.8,/,
        1 T40,'DR SYSTEM BASE REG (PHY ADDR)')

C
C       Output the DR750 Global Base Register
C
        Call LINCHK (lun,2)
        Write (lun,210) dr_gbr
210     Format (' ',T8,'DR GBR',T24,Z8.8,/,
        1 T40,'DR GLOBAL BASE REG (VIRT ADDR)')

C
C       Output the DR750 Command Block Information
C
        Call LINCHK (lun,3)
        Write (lun,220)
220     Format (/' ','DR CURRENT COMMAND BLOCK',/)

        Do 240 I = 1,3

        Call LINCHK (lun,2)
        Write (lun,230) I,dr_cb(I),v1dr_cb(I)
230     Format (' ',T8,'DR CB',I1,T24,Z8.8,/,
        1 T40,A<COMPRESSC (v1dr_cb(I))>)

240     Continue

C
C       Output the DR750 Buffer Block Information
C
        Call LINCHK (lun,3)
        Write (lun,250)
250     Format (/' ','DR CURRENT BUFFER BLOCK',/)

        Do 270 I = 1,3

        Call LINCHK (lun,2)
        Write (lun,260) I,dr_bb(I),v1dr_bb(I)
260     Format (' ',T8,'DR BB',I1,T24,Z8.8,/,
        1 T40,A<COMPRESSC (v1dr_bb(I))>)

270     Continue

C
C       Output the remaining portion of the WCS contents
C

        Call LINCHK (lun,3)
        Write (lun,280)
280     Format (/' ','DR GENERAL PURPOSE REGISTERS',/)

        Call LINCHK (lun,2)
        Write (lun,290) dr_gpr(0)
290     Format (' ',T8,'DR R0',T24,Z8.8,/,
        1 T40,'VIRTUAL ADDRESS')

        Call LINCHK (lun,2)
```

```
            Write (lun,295) dr_gpr(1)
295         Format (' ',T8,'DR R1',T24,Z8.8,/,
            1 T40,'PHYSICAL ADDRESS')

            Field = LIB$EXTZV (0,16,dr_gpr(2))

            Call LINCHK (lun,2)
            Write (lun,300) field
300         Format (' ',T8,'DR R2',T24,Z4.4,/,
            1 T40,'DEVICE STATUS REGISTER')

            Field = LIB$EXTZV (16,16,dr_gpr(2))

            Call LINCHK (lun,2)
            Write (lun,305) field
305         Format (T24,Z4.4,/,T40,'AREG')

            Call LINCHK (lun,2)
            Write (lun,310) dr_gpr(3)
310         Format (' ',T8,'DR R3',T24,Z8.8,/,
            1 T40,'CONTROL INTERCONNECT STATUS')

            Call LINCHK (lun,2)
            Write (lun,315) dr_gpr(4)
315         Format (' ',T8,'DR R4',T24,Z8.8,/,
            1 T40,'BUFFER ADDRESS')

            Call LINCHK (lun,2)
            Write (lun,320) dr_gpr(5)
320         Format (' ',T8,'DR R5',T24,Z8.8,/,
            1 T40,'RETURN ADDRESS, TRUE TEST RESULT')

            Call LINCHK (lun,6)
            Do 330, I = 6,8

            Write (lun,325) I,dr_gpr(I)
325         Format (' ',T8,'DR R',I1,T24,Z8.8,/,
            1 T40,'GENERAL PURPOSE REGISTER')

330         Continue

            Field = LIB$EXTZV (0,16,dr_gpr(9))

            Call LINCHK (lun,2)
            Write (lun,335) field
335         Format (' ',T8,'DR R9',T24,Z4.4,/,
            1 T40,'ADDRESS OF CURRENT BUFFER')

            Field = LIB$EXTZV (16,16,dr_gpr(9))

            Call LINCHK (lun,2)
            Write (lun,340) field
340         Format (T24,Z4.4,/,T40,'CURRENT BUFFER STATE')

            Field = LIB$EXTZV (0,16,dr_gpr(10))

            Call LINCHK (lun,2)
```

```
            Write (lun,345) field
345         Format (' ',T8,'DR R10',T24,Z4.4,/,
            1 T40,'ADDRESS OF NEXT BUFFER')

            Field = LIB$EXTZV (16,16,dr_gpr(10))

            Call LINCHK (lun,2)
            Write (lun,350) field
350         Format (T24,Z4.4,/,T40,'NEXT BUFFER STATUS')

            Field = LIB$EXTZV (0,16,dr_gpr(11))

            Call LINCHK (lun,2)
            Write (lun,355) field
355         Format (' ',T8,'DR R11',T24,Z4.4,/,
            1 T40,'ADDRESS OF FREE BUFFER')

            Field = LIB$EXTZV(16,16,dr_gpr(11))

            Call LINCHK (lun,2)
            Write (lun,360) field
360         Format (T24,Z4.4,/,T40,'FREE BUFFER STATUS')

            Call LINCHK (lun,2)

            Write (lun,365) dr_gpr(12)
365         Format (' ',T8,'DR R12',T24,Z8.8,/,
            1 T40,'REMAINDER OF DI BYTE COUNT')

            Call LINCHK (lun,2)
            Write (lun,370) dr_gpr(13)
370         Format (' ',T8,'DR R13',T24,Z8.8,/,
            1 T40,'CONTROL INTERCONNECT BYTE COUNT')

            Call LINCHK (lun,2)
            Write (lun,375) dr_gpr(14)
375         Format (' ',T8,'DR R14',T24,Z8.8,/,
            1 T40,'CONTROL INTERCONNECT VA')

            Call LINCHK (lun,2)
            Write (lun,380) dr_gpr(15)
380         Format (' ',T8,'DR R15',T24,Z8.8,/,
            1 T40,'CONTROL INTERCONNECT PA')

            Call LINCHK (lun,2)
            Write (lun,385) dr_gpr(16)
385         Format (' ',T8,'DR R16',T24,Z8.8,/,
            1 T40,'SILO MASK')

            Call LINCHK (lun,2)
            Write (lun,390) dr_gpr(17)
390         Format (' ',T8,'DR R17',T24,Z8.8,/,
            1 T40,'CURRENTLY EXECUTING SILO COMMAND')

            Call LINCHK (lun,2)
            Write (lun,400) dr_gpr(18)
400         Format (' ',T8,'DR R18',T24,Z8.8,/,
```

```
            1 T40,'LAST VALUE LOADED, DI BYTE COUNT')

            Call LINCHK (lun,2)
            Write (lun,405) dr_gpr(19)
    405     Format (' ',T8,'DR R19',T24,Z8.8,/,
            1 T40,'LAST VALUE LOADED, CMI BYTE COUNT')

            Call LINCHK (lun,2)
            Write (lun,410) dr_gpr(20)
     410    Format (' ',T8,'DR R20',T24,Z8.8,/,
            1 T40,'CONTROL INTERCONNECT DATA WORD')

            Call LINCHK (lun,2)
            Write (lun,415) dr_gpr(21)
    415     Format (' ',T8,'DR R21',T24,Z8.8,/,
            1 T40,'CONTROL INTERCONNECT DATA')

            Call LINCHK (lun,2)
            Write (lun,420) dr_gpr(22)
    420     Format (' ',T8,'DR R22',T24,Z8.8,/,
            1 T40,'CONTROL INTERCONNECT INFORMATION')

            call linchk (lun,1)

            write(lun,425)
    425     format(' ',:)

            call orb$l_owner (lun,emb$l_dv_ownuic)

            call ucb$l_char (lun,emb$l_dv_char)

            call ucb$w_sts (lun,emb$w_dv_sts)

            call ucb$l_opcnt (lun,emb$l_dv_opcnt)

            call ucb$w_errcnt (lun,emb$w_dv_errcnt)

            if (emb$w_hd_entry .ne. 98) then

            call linchk (lun,1)

            write(lun,425)

            call xfdriver_qio (lun,emb$w_dv_func)

            call irp$w_bcnt (lun,emb$w_dv_bcnt)

            call irp$w_boff (lun,emb$w_dv_boff)

            call irp$l_pid (lun,emb$l_dv_rqpid)

            call irp$q_iosb (lun,emb$l_dv_iosb1)
            endif

            Return
            End
```

PROGRAM SECTIONS

	Name	Bytes	Attributes
0	$CODE	3498	PIC CON REL LCL SHR EXE RD NOWRT LONG
1	$PDATA	2066	PIC CON REL LCL SHR NOEXE RD NOWRT LONG
2	$LOCAL	1668	PIC CON REL LCL NOSHR NOEXE RD WRT LONG
3	DR32	600	PIC OVR REL GBL SHR NOEXE RD WRT LONG
4	EMB	512	PIC OVR REL GBL SHR NOEXE RD WRT LONG
	Total Space Allocated	8344	

ENTRY POINTS

Address	Type	Name
0-00000000		DR750

VARIABLES

Address	Type	Name	Address	Type	Name
4-00000066	I*4	DR_CMI_ADR	4-0000006A	I*4	DR_CMI_BCR
4-00000052	I*4	DR_DCR	4-0000006E	I*4	DR_DI_BCR
4-00000072	I*4	DR_DSL	4-0000007A	I*4	DR_GBR
4-00000076	I*4	DR_SBR	4-00000056	I*4	DR_UTL
4-0000005A	I*4	DR_WCS_ADR	4-0000005E	I*4	DR_WCS_DAT
4-0000001C	L*1	EMB$B_DV_CLASS	4-00000010	L*1	EMB$B_DV_ERTCNT
4-00000011	L*1	EMB$B_DV_ERTMAX	4-0000003E	L*1	EMB$B_DV_NAMLNG
4-0000003A	L*1	EMB$B_DV_SLAVE	4-0000001D	L*1	EMB$B_DV_TYPE
4-00000036	I*4	EMB$L_DV_CHAR	4-00000012	I*4	EMB$L_DV_IOSB1
4-00000016	I*4	EMB$L_DV_IOSB2	4-00000026	I*4	EMB$L_DV_MEDIA
4-0000004E	I*4	EMB$L_DV_NUMREG	4-0000002E	I*4	EMB$L_DV_OPCNT
4-00000032	I*4	EMB$L_DV_OWNUIC	4-0000001E	I*4	EMB$L_DV_RQPID
4-00000000	I*4	EMB$L_HD_SID	4-0000003F	CHAR	EMB$T_DV_NAME
4-00000024	I*2	EMB$W_DV_BCNT	4-00000022	I*2	EMB$W_DV_BOFF
4-0000002C	I*2	EMB$W_DV_ERRCNT	4-0000003C	I*2	EMB$W_DV_FUNC
4-0000001A	I*2	EMB$W_DV_STS	4-0000002A	I*2	EMB$W_DV_UNIT
4-00000004	I*2	EMB$W_HD_ENTRY	4-0000000E	I*2	EMB$W_HD_ERRSEQ
2-000002AC	I*4	FIELD	2-000002B0	I*4	FIELDT
2-000002B4	R*4	FLOAT	2-000002B8	I*4	I
AP-00000004@	L*1	LUN	2-000002A8	I*4	SILO_BYTE_CNT

ARRAYS

Address	Type	Name	Bytes	Dimensions
2-000001E7	CHAR	BUS_FUNC	192	(0:7)
4-0000008A	I*4	DR_BB	12	(3)
4-0000007E	I*4	DR_CB	12	(3)
2-00000000	CHAR	DR_DCR1	48	(11:12)
2-00000030	CHAR	DR_DCR2	119	(14:20)
2-000000A7	CHAR	DR_DCR3	320	(22:31)
4-00000096	I*4	DR_GPR	92	(0:22)

```
4-00000000  L*1  EMB                          512  (0:511)
4-00000052  I*4  EMB$L_DV_REGSAV              420  (0:104)
4-00000006  I*4  EMB$Q_HD_TIME                  8  (2)
3-0000019B  CHAR V1DR_BB                       93  (3)
3-000001F8  CHAR V1DR_CB                       96  (3)
3-00000000  CHAR V1DR_SL                      345  (0:14)
3-00000159  CHAR V2DR_SL                       66  (21:23)
```

LABELS

Address	Label	Address	Label	Address	Label	Address	Label	Address	Label	Address	Label
1-00000054	20'	1-00000068	25'	1-00000081	30'	1-00000099	40'	1-000000AC	45'	1-000000D1	50'
1-000000F5	60'	1-00000118	70'	1-0000013B	80'	1-00000167	85'	1-00000190	100'	1-000001A2	105'
1-000001C5	110'	1-000001E6	150'	1-0000020B	160'	1-0000023E	170'	1-00000251	175'	1-00000274	180'
**	185	1-0000029F	190'	1-000002A9	200'	1-000002DE	210'	1-00000314	220'	1-00000334	230'
**	240	1-00000351	250'	1-00000370	260'	**	270	1-0000038D	280'	1-000003B1	290'
1-000003D7	295'	1-000003FE	300'	1-0000042B	305'	1-0000043A	310'	1-0000046C	315'	1-00000491	320'
1-000004C8	325'	**	330	1-000004F8	335'	1-00000528	340'	1-00000547	345'	1-00000575	350'
1-00000592	355'	1-000005C0	360'	1-000005DD	365'	1-0000060F	370'	1-00000646	375'	1-00000675	380'
1-000006A4	385'	1-000006C5	390'	1-000006FD	400'	1-00000735	405'	1-0000076E	410'	1-000007A4	415'
1-000007D5	420'	1-0000080D	425'								

FUNCTIONS AND SUBROUTINES REFERENCED

Type	Name	Type	Name	Type	Name	Type	Name	Type	Name	Type	Name
I*4	COMPRESS4	I*4	COMPRESSC	I*4	COMPRESSF		DHEAD1		FRCTOF		IRP$L_PID
	IRP$Q_IOSB		IRP$W_BCNT		IRP$W_BOFF	I*4	LIB$EXTZV		LIB$INSV		LINCHR
	ORB$L_OWNER		OUTPUT		UCB$L_CHAR		UCB$L_OPCNT		UCB$W_ERRCNT		UCB$W_STS
	XFDRIVER_QIO										

```
0001
0002


COMMAND QUALIFIERS

  FORTRAN /LIS=LIS$:DR750/OBJ=OBJ$:DR750 MSRC$:DR750

  /CHECK=(NOBOUNDS,OVERFLOW,NOUNDERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /STANDARD=(NOSYNTAX,NOSOURCE_FORM)
  /SHOW=(NOPREPROCESSOR,NOINCLUDE,MAP)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOCROSS_REFERENCE  /NOMACHINE_CODE  /CONTINUATIONS=19


COMPILATION STATISTICS

  Run Time:           9.31 seconds
  Elapsed Time:       20.52 seconds
  Page Faults:        262
  Dynamic Memory:     251 pages
```

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

CLASSIFY
LIS

DR750
LIS

CSTRING
LIS

DHEADS
LIS

DR780
LIS

DR11W
LIS

COMPRESS
LIS

DTAILS
LIS

DECODECC
LIS

DUMPREG
LIS

DUTLDRIVR
LIS

CALCMAP
LIS

DUP3271
LIS

CRYPTK
LIS

DQDISKS
LIS

DUP11
LIS