```
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  EEEEEEEEEEE      RRRRRRRRRRRR
DDD          DDD RRRRRRRRRRRR           III       VVV         VVV  EEEEEEEEEEE      RRRRRRRRRRRR
DDD          DDD RRRRRRRRRRRR           III       VVV         VVV  EEEEEEEEEEE      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
```

```
DDDDDDDD    LL          DDDDDDDD    RRRRRRRR      IIIIII    VV      VV  EEEEEEEEEE  RRRRRRRR
DDDDDDDD    LL          DDDDDDDD    RRRRRRRR      IIIIII    VV      VV  EEEEEEEEEE  RRRRRRRR
DD      DD  LL          DD      DD  RR      RR      II      VV      VV  EE          RR      RR
DD      DD  LL          DD      DD  RR      RR      II      VV      VV  EE          RR      RR
DD      DD  LL          DD      DD  RR      RR      II      VV      VV  EE          RR      RR
DD      DD  LL          DD      DD  RR      RR      II      VV      VV  EE          RR      RR
DD      DD  LL          DD      DD  RRRRRRRR        II      VV      VV  EEEEEEEE    RRRRRRRR
DD      DD  LL          DD      DD  RRRRRRRR        II      VV      VV  EEEEEEEE    RRRRRRRR
DD      DD  LL          DD      DD  RR  RR          II      VV      VV  EE          RR  RR
DD      DD  LL          DD      DD  RR  RR          II      VV      VV  EE          RR  RR
DD      DD  LL          DD      DD  RR    RR        II        VV  VV    EE          RR    RR
DD      DD  LL          DD      DD  RR    RR        II        VV  VV    EE          RR    RR
DDDDDDDD    LLLLLLLLLL  DDDDDDDD    RR      RR    IIIIII        VV      EEEEEEEEEE  RR      RR
DDDDDDDD    LLLLLLLLLL  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

(1)	113	EXTERNAL AND LOCAL DEFINITIONS
(1)	299	STANDARD TABLES
(1)	522	CONTROLLER INITIALIZATION ROUTINE
(1)	572	UNIT INITIALIZATION ROUTINE
(1)	695	DRIVER SPECIFIC SUBROUTINES
(1)	718	FDT ROUTINE - TEST TRANSFER BYTE COUNT ALIGNMENT
(1)	754	START I/O ROUTINE
(1)	1532	INTERRUPT SERVICE ROUTINE
(1)	1593	REGISTER DUMP ROUTINE
(1)	1632	MOVE TO USER BUFFER ROUTINE
(1)	1676	MOVE FROM USER BUFFER ROUTINE

```
0000     1          .TITLE  DLDRIVER - VAX/VMS RL11/RL01,RL02 DISK DRIVER
0000     2          .IDENT  'V04-000'
0000     3 ;
0000     4 ;****************************************************************************
0000     5 ;*                                                                          *
0000     6 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000     7 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000     8 ;*  ALL RIGHTS RESERVED.                                                    *
0000     9 ;*                                                                          *
0000    10 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000    11 ;*  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE          *
0000    12 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER     *
0000    13 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000    14 ;*  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY       *
0000    15 ;*  TRANSFERRED.                                                            *
0000    16 ;*                                                                          *
0000    17 ;*  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE    *
0000    18 ;*  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT        *
0000    19 ;*  CORPORATION.                                                            *
0000    20 ;*                                                                          *
0000    21 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS     *
0000    22 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000    23 ;*                                                                          *
0000    24 ;*                                                                          *
0000    25 ;****************************************************************************
0000    26 ;
0000    27 ;
0000    28 ; FACILITY:
0000    29 ;
0000    30 ;       VAX/VMS RL11/RL01,RL02 DISK DRIVER
0000    31 ;
0000    32 ; AUTHOR:
0000    33 ;
0000    34 ; C. FRANKS     05-OCT-1979
0000    35 ;
0000    36 ; MODIFIED BY:
0000    37 ;
0000    38 ;       V03-008 WHM0001         Bill Matthews           15-May-1984
0000    39 ;               Added MicroVAX I/QBUS support.
0000    40 ;
0000    41 ;       V03-007 RAS0300         Ron Schaefer            27-Apr-1984
0000    42 ;               Add DEV$M_NNM characteristic to DECHAR2 so that these
0000    43 ;               devices will have the "node$" prefix.
0000    44 ;
0000    45 ;       V03-006 PRD0033         Paul R. DeStefano       09-Sep-1983
0000    46 ;               Added EXE$LCLDSKVALID to function decision table.
0000    47 ;
0000    48 ;       V03-005 ROW0211         Ralph O. Weber          16-AUG-1983
0000    49 ;               Change device-dependent UCB definition base from UCB$W_BCR+2
0000    50 ;               to UCB$K_LCL_DISK_LENGTH.
0000    51 ;
0000    52 ;       V03-004 KDM0059         Kathleen D. Morse       14-Jul-1983
0000    53 ;               Change time-wait loops to use new TIMEDWAIT macro.
0000    54 ;
0000    55 ;       V03-003 PRD0020         Paul R. DeStefano       26-Apr-1983
0000    56 ;               Modified FATALERR routine to return SS$_PARITY only for
0000    57 ;               errors that possibly indicate bad media.  All other error
```

```
0000     58 ;                  conditions whic'  ormerly returned SS$_PARITY now return
0000     59 ;                  SS$_CNTLERR.
0000     60 ;
0000     61 ;        V03-002 KDM0002         Kathleen D. Morse       28-Jun-1982
0000     62 ;                Added $DYNDEF.
0000     63 ;
0000     64 ;        V03-001 KTA0100         Kerbey T. Altmann       07-Jun-1982
0000     65 ;                Add code to set UCB$L_MEDIA_ID.
0000     66 ;
0000     67 ;**
```

```
0000     69 ; ABSTRACT:
0000     70 ;
0000     71 ;       THIS MODULE CONTAINS THE TABLES AND ROUTINES NECESSARY TO
0000     72 ;       PERFORM ALL DEVICE-DEPENDENT PROCESSING OF AN I/O REQUEST
0000     73 ;       FOR RL11/RL01,RL02 DISK TYPES ON A VAX/VMS SYSTEM.
0000     74 ;
0000     75 ;       THE DISKS HAVE THE FOLLOWING PHYSICAL GEOMETRY:
0000     76 ;
0000     77 ;                          TRACKS/     SECTORS/      BYTES/      MAXIMUM
0000     78 ;             # CYL        CYLINDER    TRACK         SECTOR      BLOCKS
0000     79 ;
0000     80 ; RL01         256           2           40            256        10240
0000     81 ; RL02         512           2           40            256        20480
0000     82 ;
0000     83 ;       SINCE THE SECTOR SIZE IS ONLY 1/2 BLOCK, LOGICAL TO PHYSICAL
0000     84 ;       CONVERSION OF THE DISK ADDRESS IS DONE IN THE DRIVER STARTIO
0000     85 ;       ROUTINE RATHER THAN IN THE IOC$CVTLOGPHY FDT ROUTINE.
0000     86 ;
0000     87 ;       OVERLAPPED SEEKS ARE NOT ATTEMPTED BECAUSE THE DEVICE DOES
0000     88 ;       NOT INTERRUPT AT THE COMPLETION OF A SEEK.
0000     89 ;
0000     90 ;       ALSO, THE DEVICE DOES NOT PERFORM AN IMPLICIT SEEK WHEN PERFORMING
0000     91 ;       A READ OR WRITE FUNCTION,SO SEEK FUNCTIONS ARE ISSUED BY THIS
0000     92 ;       DRIVER WHERE NECESSARY PRIOR TO ISSUING A READ OR WRITE FUNCTION.
0000     93 ;       THE READ OR WRITE FUNCTION IS THEN ISSUED AS SOON AS THE RL11
0000     94 ;       CONTROLLER COMES READY (WHILE THE SEEK IS IN PROGRESS), AND A
0000     95 ;       WAIT FOR INTERRUPT (UPON COMPLETION OF THE READ OR WRITE) IS
0000     96 ;       ISSUED.  IF A SEEK FUNCTION IS REQUESTED SEPARATELY FROM A READ OR
0000     97 ;       WRITE, A DUMMY READ HEADER FUNCTION IS ISSUED FOLLOWING THE SEEK
0000     98 ;       FUNCTION AND A WAIT FOR INTERRUPT (UPON COMPLETION OF THE READ
0000     99 ;       HEADER) IS ISSUED.
0000    100 ;
0000    101 ;       THE IO$X_INHSEEK FUNCTION MODIFIER IS TREATED AS A NO-OP BY
0000    102 ;       THIS DRIVER, SINCE AN EXPLICIT SEEK IS NECESSARY FOR THE RL02
0000    103 ;       TO TRANSFER DATA PROPERLY.
0000    104 ;
0000    105 ;       THE RL'S DO NOT READ OR WRITE BEYOND THE END OF TRACK (THEY DO NOT
0000    106 ;       AUTOMATICALLY SEEK THE NEXT TRACK), SO  ALL READ AND WRITE FUNCTIONS
0000    107 ;       ARE BROKEN UP BY THIS DRIVER INTO PARTIAL TRANSFERS TO THE END OF
0000    108 ;       TRACK, FOLLOWED BY A SEEK TO THE NEXT TRACK, THEN ANOTHER READ OR
0000    109 ;       WRITE FUNCTION UNTIL THE TOTAL DATA TRANSFER IS COMPLETE.
0000    110 ;
0000    111 ;--
```

```
0000   113          .SBTTL  EXTERNAL AND LOCAL DEFINITIONS
0000   114
0000   115 ;
0000   116 ; EXTERNAL SYMBOLS
0000   117 ;
0000   118
0000   119          $ADPDEF                         ;DEFINE ADAPTER CONTROL BLOCK
0000   120          $CRBDEF                         ;DEFINE CHANNEL REQUEST BLOCK
0000   121          $DCDEF                          ;DEFINE DEVICE CLASS
0000   122          $DDBDEF                         ;DEFINE DEVICE DATA BLOCK
0000   123          $DEVDEF                         ;DEFINE DEVICE CHARACTERISTICS
0000   124          $DPTDEF                         ;DEFINE DRIVER PROLOGUE TABLE
0000   125          $DYNDEF                         ;DEFINE DYNAMIC DATA STRUCTURE TYPES
0000   126          $EMBDEF                         ;DEFINE ERROR MESSAGE BUFFER
0000   127          $IDBDEF                         ;DEFINE INTERRUPT DATA BLOCK
0000   128          $IODEF                          ;DEFINE I/O FUNCTION CODES
0000   129          $IRPDEF                         ;DEFINE I/O REQUEST PACKET
0000   130          $PRDEF                          ;DEFINE PROCESSOR REGISTERS
0000   131          $PTEDEF                         ;DEFINE SYSTEM PTES
0000   132          $SSDEF                          ;DEFINE SYSTEM STATUS CODES
0000   133          $UCBDEF                         ;DEFINE UNIT CONTROL BLOCK
0000   134          $VADEF                          ;DEFINE VIRTUAL ADDRESS BITS
0000   135          $VECDEF                         ;DEFINE INTERRUPT VECTOR BLOCK
0000   136
0000   137 ;
0000   138 ; LOCAL MACROS
0000   139 ;
0000   140
0000   141 ;        EXFUNCL
0000   142 ;        BRANCH TO SUBROUTINE WHICH REQUESTS CHANNEL (IF NOT ALREADY OWNED),
0000   143 ;        EXECUTES FCODE (OR R3) FUNCTION, AND BRANCHES TO BDST ON ERROR
0000   144
0000   145          .MACRO  EXFUNCL BDST,FCODE
0000   146                  .IF NB  FCODE           ;IS FCODE NON-BLANK?
0000   147                  MOVZBL  #CD'FCODE,R3    ;IF NB - SPECIFY FCODE FUNCTION
0000   148                  .ENDC                   ;IF B - SPECIFY FNTN IN EXISTING R3
0000   149                  BSBW    FEXL            ;EXECUTE FUNCTION
0000   150                  .BYTE   BDST-.-1        ;WHERE TO GO IF ERROR
0000   151          .ENDM
0000   152
0000   153
0000   154 ;        GENF
0000   155 ;        GENERATE FUNCTION TABLE ENTRY AND CASE TABLE INDEX SYMBOL
0000   156
0000   157          .MACRO  GENF FCODE
0000   158                  CD'FCODE=.-FTAB/2
0000   159                  .WORD   FCODE!RL_CS_M_IE  ;FCODE WITH INT ENABLE BIT
0000   160          .ENDM
0000   161
0000   162
0000   163 ;        CKPWR
0000   164 ;        DISABLE INTERRUPTS, CHECK IF POWER HAS FAILED,
0000   165 ;        AND PUT DEVICE UNIT NUMBER IN R2<9:8>
0000   166 ;
0000   167
0000   168          .MACRO  CKPWR ?L1
0000   169                  CLRL    R2              ;CLEAR R2 FOR UNIT NUMBER
```

```
                     0000   170                 INSV    UCB$W_UNIT(R5),-  ;PUT UNIT # IN R2<9:8>
                     0000   171                         #8,#2,R2          ;...
                     0000   172                 DSBINT                    ;DISABLE INTERRUPTS
                     0000   173                 BBC     #UCB$V_POWER,-    ;IF CLR - NO POWER FAILURE
                     0000   174                         UCB$W_STS(R5),L1  ;...
                     0000   175                 ENBINT                    ;POWER FAILURE - ENABLE INTERRUPTS
                     0000   176                 BRW     RETREG            ;EXIT
                     0000   177 L1:                                       ;RETURN FOR NO POWER FAILURE
                     0000   178         .ENDM
                     0000   179
                     0000   180
                     0000   181 ;
                     0000   182 ; LOCAL SYMBOLS
                     0000   183 ;
                     0000   184
00000004             0000   185 RL_NUM_REGS     =4                        ;NUMBER OF DEVICE REGISTERS
00000005             0000   186 RL_SLM          =5                        ;STATE=SEEK LINEAR MODE (READY TO GO)
000000C9             0000   187 UCB$B_DL_DCHEK  =UCB$W_OFFSET+1           ;REDEFINE FOR DATA CHECK USE
                     0000   188
                     0000   189 ;
                     0000   190 ; UCB OFFSETS WHICH FOLLOW THE STANDARD UCB FIELDS
                     0000   191 ;
                     0000   192         $DEFINI UCB                       ;START OF UCB DEFINITIONS
                     0000   193
000000CC             0000   194 .=UCB$K_LCL_DISK_LENGTH                    ;BEGIN DEFINITIONS AT END OF UCB
                     00CC   195 $DEF     UCB$W_DL_PBCR    .BLKW   1        ;PARTIAL BYTE COUNT
                     00CE   196 $DEF     UCB$W_DL_CS      .BLKW   1        ;CONTROL STATUS REGISTER
                     00D0   197 $DEF     UCB$W_DL_BA      .BLKW   1        ;BUS ADDRESS REGISTER
                     00D2   198 $DEF     UCB$W_DL_DA      .BLKW   1        ;DISK ADDRESS REGISTER
                     00D4   199 $DEF     UCB$W_DL_MP      .BLKW   1        ;MULTIPURPOSE REGISTER
                     00D6   200 $DEF     UCB$W_DL_DPN     .BLKW   1        ;DATA PATH NUMBER
                     00D8   201 $DEF     UCB$L_DL_SVAPTE                   ;SAVED SVAPTE OF THE USER'S BUFFER
                     00D8   202 $DEF     UCB$L_DL_DPR     .BLKL   1        ;DATAPATH REGISTER
                     00DC   203 $DEF     UCB$L_DL_BUFADR                   ;USER BUFFER ADDRESS
                     00DC   204 $DEF     UCB$L_DL_FMPR    .BLKL   1        ;FINAL MAP REGISTER
                     00E0   205 $DEF     UCB$A_DL_MOVRTN                   ;BUFFER MOVE ROUTINE ADDRESS
                     00E0   206 $DEF     UCB$L_DL_PMPR    .BLKL   1        ;PREVIOUS MAP REGISTER
                     00E4   207 $DEF     UCB$B_DL_DPPE    .BLKB   1        ;DATAPATH PURGE ERROR
                     00E5   208 $DEF     UCB$W_DL_DB      .BLKW   3        ;DATA BUFFER REGISTER
                     00EB   209 $DEF     UCB$B_DL_XBA     .BLKB   1        ;BUS ADDRESS EXTENSION BITS
                     00EC   210 $DEF     UCB$W_DL_SBA     .BLKW   1        ;SAVED BUFFER ADDRESS
                     00EE   211 $DEF     UCB$A_DL_BUF_VA  .BLKL   1        ;PHYSICAL BUFFER VIRTUAL ADDRESS
                     00F2   212 $DEF     UCB$A_DL_BUF_PA  .BLKL   1        ;PHYSICAL BUFFER PHYSICAL ADDRESS
                     00F6   213 $DEF     UCB$W_DL_FLAGS   .BLKW   1        ;FLAGS
                     00F8   214         $VIELD  UCB,0,<-                  ;START THE FLAG DEFINITIONS
                     00F8   215                 <DL_22BIT,,M>,-           ;22 BIT ADDRESSING
                     00F8   216                 <DL_MAPPING,,M>,-         ;ADAPTER MAPPING
                     00F8   217                 >                         ;END OF FLAG DEFINITIONS
                     00F8   218 $DEF     UCB$K_DL_LEN     .BLKW   1        ;LENGTH OF UCB
                     00FA   219 $EQU     UCB$K_DL_BUFSZ 20                 ;BUFFER SIZE = 40 SECTORS *
                     00FA   220                                            ;256 BYTES/SECTOR / 512 BYTES/PAGE
                     00FA   221         $DEFEND UCB                       ;END OF UCB DEFINITONS
                     0000   222
                     0000   223 ;
                     0000   224 ; RL11/RL01 REGISTER OFFSETS FROM CSR ADDRESS
                     0000   225 ;
                     0000   226         $DEFINI RL                        ; START OF REGISTER DEFINITIONS
```

```
                     0000   227
                     0000   228 $DEF     RL_CS           .BLKW   1       ;CONTROL STATUS REGISTER (CSR)
                     0002   229             _VIELD  RL_CS,0,<-              ;START OF CSR BIT DEFINITIONS
                     0002   230                     <DRDY,,M>,-             ; DRIVE READY
                     0002   231                     <FCODE,3>,-             ; FUNCTION CODE
                     0002   232                     <XBA,2>,-               ; BUS ADDRESS EXTENSION BITS
                     0002   233                     <IE,,M>,-               ; INTERRUPT ENABLE
                     0002   234                     <CRDY,,M>,-             ; CONTROLLER READY
                     0002   235                     <DS,2>,-                ; DRIVE SELECT
                     0002   236                     <OPI,,M>,-              ; OPERATION INCOMPLETE
                     0002   237                     <CRC,,M>,-              ; DATA CRC OR HEADER CRC
                     0002   238                     <DLT,,M>,-              ; DATA LATE OR HEADER NOT FOUND
                     0002   239                     <NXM,,M>,-              ; NON-EXISTENT MEMORY
                     0002   240                     <DE,,M>,-               ; DRIVE ERROR
                     0002   241                     <CE,,M>-                ; COMPOSITE ERROR
                     0002   242             >                               ;END CSR BIT DEFINITIONS
                     0002   243
                     0002   244 $DEF     RL_BA           .BLKW   1       ;BUS ADDRESS REGISTER (BAR)
                     0004   245
                     0004   246 $DEF     RL_DA           .BLKW   1       ;DISK ADDRESS REGISTER (DAR)
                     0006   247             _VIELD  RL_DA,0,<-              ;START OF DAR BIT DEFINITIONS
                     0006   248                     <MRK,,M>,-              ; MARK (ALWAYS 1)
                     0006   249                     <STS,,M>,-              ; GET STATUS
                     0006   250                     <,1>,-                  ; RESERVED BIT
                     0006   251                     <RST,,M>,-              ; RESET
                     0006   252                     <,12>,-                 ; RESERVED BITS
                     0006   253             >                               ;END OF DAR BIT DEFINITIONS
                     0006   254
                     0006   255 $DEF     RL_MP           .BLKW   1       ;MULTIPURPOSE REGISTER (MPR)
                     0008   256             _VIELD  RL_MP,0,<-              ;START OF MPR BIT DEFINITIONS
                     0008   257                     <STA,3>,-               ; DRIVE STATE
                     0008   258                     <BH,,M>,-               ; BRUSH HOME
                     0008   259                     <HO,,M>,-               ; HEADS OUT
                     0008   260                     <CO,,M>,-               ; COVER OPEN
                     0008   261                     <HS,,M>,-               ; HEAD SELECT
                     0008   262                     <TYP,,M>,-              ; DRIVE TYPE
                     0008   263                     <DSE,,M>,-              ; DRIVE SELECT ERROR
                     0008   264                     <VC,,M>,-               ; VOLUME CHECK
                     0008   265                     <WGE,,M>,-              ; WRITE GATE ERROR
                     0008   266                     <SPE,,M>,-              ; SPIN ERROR
                     0008   267                     <SKTO,,M>,-             ; SEEK TIME OUT
                     0008   268                     <WL,,M>,-               ; WRITE LOCK
                     0008   269                     <CHE,,M>,-              ; CURRENT HEAD ERROR
                     0008   270                     <WDE,,M>-               ; WRITE DATA ERROR
                     0008   271             >                               ;END MPR BIT DEFINITIONS
                     0008   272
                     0008   273 $DEF     RL_BAE          .BLKW   1       ; BUS ADDRESS EXTENSION REGISTER(BAE)
                     000A   274
                     000A   275             $DEFEND RL                      ;END RL11/RL01 REGISTER DEFINITIONS
                     0000   276
                     0000   277 ;
                     0000   278 ; HARDWARE FUNCTION CODES
                     0000   279 ;
00000000             0000   280 F_NOP=0*2                                   ;NO OPERATION
00000000             0000   281 F_UNLOAD=F_NOP                              ;NO OPERATION
00000006             0000   282 F_SEEK=3*2                                  ;SEEK CYLINDER
00000000             0000   283 F_RECAL=F_NOP                               ;NO OPERATION
```

```
00000004  0000   284 F_DRVCLR=2*2                    ;DRIVE CLEAR (GET STATUS)
00000000  0000   285 F_RELEASE=F_NOP                 ;NO OPERATION
00000000  0000   286 F_OFFSET=F_NOP                  ;NO OPERATION
00000000  0000   287 F_RETCENTER=F_NOP               ;NO OPERATION
00000004  0000   288 F_PACKACK=2*2                   ;PACK ACKNOWLEDGE (SET VOLUME VALID)
00000000  0000   289 F_SEARCH=F_NOP                  ;NO OPERATION
00000002  0000   290 F_WRITECHECK=1*2                ;WRITE CHECK
0000000A  0000   291 F_WRITEDATA=5*2                 ;WRITE DATA
00000000  0000   292 F_WRITEHEAD=F_NOP               ;NO OPERATION
0000000C  0000   293 F_READDATA=6*2                  ;READ DATA
00000008  0000   294 F_READHEAD=4*2                  ;READ HEADER
00000000  0000   295 F_AVAILABLE=F_NOP               ;NO OPERATION
00000004  0000   296 F_GETSTATUS=2*2                 ;GET STATUS (DRIVER INTERNAL USE)
          0000   297
```

```
                        .SBTTL  STANDARD TABLES

;
; DRIVER PROLOGUE TABLE
;
;       THE DPT DESCRIBES DRIVER PARAMETERS AND I/O DATABASE FIELDS
;       THAT ARE TO BE INITIALIZED DURING DRIVER LOADING AND RELOADING
;

        DPTAB   -                               ;DPT CREATION MACRO
                END=DL_END,-                    ;END OF DRIVER LABEL
                ADAPTER=UBA,-                   ;ADAPTER TYPE = UNIBUS
                FLAGS=DPT$M_SVP,-               ;SYSTEM PAGE TABLE ENTRY REQUIRED
                UCBSIZE=UCB$K_DL_LEN,-          ;LENGTH OF UCB
                NAME=DLDRIVER                   ;DRIVER NAME

        DPT_STORE INIT                          ;START CONTROL BLOCK INIT VALUES
        DPT_STORE DDB,DDB$L_ACPD,L,<^A\F11\>    ;DEFAULT ACP NAME
        DPT_STORE DDB,DDB$L_ACPD+3,B,DDB$K_CART ;ACP CLASS
        DPT_STORE UCB,UCB$B_FIPL,B,8            ;FORK IPL
        DPT_STORE UCB,UCB$L_DEVCHAR,L,-         ;DEVICE CHARACTERISTICS
                <DEV$M_FOD-                     ; FILES ORIENTED
                !DEV$M_DIR-                     ; DIRECTORY STRUCTURED
                !DEV$M_AVL-                     ; AVAILABLE
                !DEV$M_ELG-                     ; ERROR LOGGING
                !DEV$M_SHR-                     ; SHAREABLE
                !DEV$M_IDV-                     ; INPUT DEVICE
                !DEV$M_ODV-                     ; OUTPUT DEVICE
                !DEV$M_RND>                     ; RANDOM ACCESS
        DPT_STORE UCB,UCB$L_DEVCHAR2,L,-        ; DEVICE CHARACTERISTICS
                <DEV$M_NNM>                     ; PREFIX NAME WITH "node$"
        DPT_STORE UCB,UCB$B_DEVCLASS,B,DC$_DISK ;DEVICE CLASS
        DPT_STORE UCB,UCB$W_DEVBUFSIZ,W,512     ;DEFAULT BUFFER SIZE
        DPT_STORE UCB,UCB$B_SECTORS,B,40        ;NUMBER OF SECTORS PER TRACK
        DPT_STORE UCB,UCB$B_TRACKS,B,2          ;NUMBER OF TRACKS PER CYLINDER
        DPT_STORE UCB,UCB$B_DIPL,B,21           ;DEVICE IPL
        DPT_STORE UCB,UCB$B_ERTMAX,B,8          ;MAX ERROR RETRY COUNT
        DPT_STORE UCB,UCB$W_DEVSTS,W,-          ;INHIBIT LOG TO PHYS CONVERSION IN FDT
                <UCB$M_NOCNVRT>                 ;...

        DPT_STORE REINIT                        ;START CONTROL BLOCK RE-INIT VALUES
        DPT_STORE CRB,CRB$L_INTD+4,D,DL_INT     ;INTERRUPT SERVICE ROUTINE ADDRESS
        DPT_STORE CRB,CRB$L_INTD+VEC$L_INITIAL,-    ;CONTROLLER INIT ADDRESS
                        D,DL_RL11_INIT          ;...
        DPT_STORE CRB,CRB$L_INTD+VEC$L_UNITINIT,-   ;UNIT INIT ADDRESS
                        D,DL_RLOX_INIT          ;...
        DPT_STORE DDB,DDB$L_DDT,D,DL$DDT        ;DDT ADDRESS

        DPT_STORE END                           ;END OF INITIALIZATION TABLE

;
; DRIVER DISPATCH TABLE
;
;       THE DDT LISTS ENTRY POINTS FOR DRIVER SUBROUTINES WHICH ARE
;       CALLED BY THE OPERATING SYSTEM.
;
```

```
                        356          DDTAB   -                              ;DDT CREATION MACRO
                        357                  DEVNAM=DL,-                    ;NAME OF DEVICE
                        358                  START=DL_STARTIO,-             ;START I/O ROUTINE
                        359                  UNSOLIC=DL_UNSOLNT,-           ;UNSOLICITED INTERRUPT
                        360                  FUNCTB=DL_FUNCTABLE,-          ;FUNCTION DECISION TABLE
                        361                  CANCEL=0,-                     ;CANCEL=NO-OP FOR FILES DEVICE
                        362                  REGDMP=DL_REGDUMP,-            ;REGISTER DUMP ROUTINE
                        363                  DIAGBF=<<RL_NUM_REGS+5+5+3+1>*4>,-  ;BYTES IN DIAG BUFFER
                        364                  ERLGBF=<<<RL_NUM_REGS+5+1>*4>+EMB$L_DV_REGSAV>  ;BYTES IN
                        365                                                 ;ERROR LOG BUFFER
                        366
                        367  ; DIAGNOSTIC BUFFER SIZE = <<4 RL02 REGISTER LONGWORDS + 5 UCB FIELD LONGWORDS
                        368  ;                          + 5 IOC$DIAGBUFILL LONGWORDS + 3 BUFFER ALLOCATION
                        369  ;                          LONGWORDS + 1 LONGWORD FOR # REGISTERS IN DL_REGDUMP>
                        370  ;                          * 4 BYTES/LONGWORD>
                        371  ;
                        372  ; ERROR LOG BUFFER SIZE  = <<<4 RL02 REGISTER LONGWORDS + 5 UCB FIELD LONGWORDS
                        373  ;                          + 1 LONGWORD FOR # REGISTERS IN DL_REGDUMP>
                        374  ;                          * 4 BYTES/LONGWORD> + BYTES NEEDED FOR ERROR LOGGER
                        375  ;                          TO SAVE SOFTWARE REGISTERS>
                        376
                        377
                        378  ;
                        379  ; HARDWARE FUNCTION CODE TABLE
                        380  ;
                        381  ;       THIS TABLE MERGES THE FUNCTION CODE BITS WITH THE
                        382  ;       INTERRUPT ENABLE BIT AND GENERATES THE CASE TABLE
                        383  ;       INDEX SYMBOL.
                        384
0038    385  FTAB:      GENF    F_NOP                  ;NO-OP
003A    386             GENF    F_UNLOAD               ;UNLOAD VOLUME (NOP)
003C    387             GENF    F_SEEK                 ;SEEK
003E    388             GENF    F_RECAL                ;RECALIBRATE (NOP)
0040    389             GENF    F_DRVCLR               ;DRIVE CLEAR (RESET & GET STATUS)
0042    390             GENF    F_RELEASE              ;RELEASE PORT (NOP)
0044    391             GENF    F_OFFSET               ;OFFSET HEADS (NOP)
0046    392             GENF    F_RETCENTER            ;RETURN HEADS TO CENTERLINE (NOP)
0048    393             GENF    F_PACKACK              ;PACK ACKNOWLEDGE (RESET & GET STATUS)
004A    394             GENF    F_SEARCH               ;SEARCH (NOP)
004C    395             GENF    F_WRITECHECK           ;WRITE CHECK
004E    396             GENF    F_WRITEDATA            ;WRITE DATA
0050    397             GENF    F_READDATA             ;READ DATA
0052    398             GENF    F_WRITEHEAD            ;WRITE HEADERS (NOP)
0054    399             GENF    F_READHEAD             ;READ HEADERS
0056    400             GENF    F_NOP                  ;place holder
0058    401             GENF    F_NOP                  ;place holder
005A    402             GENF    F_AVAILABLE            ;AVAILABLE
005C    403
```

```
                     ;
                     ; FUNCTION DECISION TABLE
                     ;
                     ;       THE FDT LISTS VALID FUNCTION CODES, SPECIFIES WHICH
                     ;       CODES ARE BUFFERED, AND DESIGNATES SUBROUTINES TO
                     ;       PERFORM PREPROCESSING FOR PARTICULAR FUNCTIONS.
                     ;

                     DL_FUNCTABLE:
                             FUNCTAB ,-                      ;LIST LEGAL FUNCTIONS
                                     <NOP,-                  ; NO-OP
                                     UNLOAD,-                ; UNLOAD
                                     SEEK,-                  ; SEEK
                                     DRVCLR,-                ; DRIVE CLEAR
                                     PACKACK,-               ; PACK ACKNOWLEDGE
                                     SENSECHAR,-             ; SENSE CHARACTERISTICS
                                     SETCHAR,-               ; SET CHARACTERISTICS
                                     SENSEMODE,-             ; SENSE MODE
                                     SETMODE,-               ; SET MODE
                                     WRITECHECK,-            ; WRITE CHECK
                                     READHEAD,-              ; READ HEADER
                                     READLBLK,-              ; READ LOGICAL BLOCK
                                     WRITELBLK,-             ; WRITE LOGICAL BLOCK
                                     READPBLK,-              ; READ PHYSICAL BLOCK
                                     WRITEPBLK,-             ; WRITE PHYSICAL BLOCK
                                     READVBLK,-              ; READ VIRTUAL BLOCK
                                     WRITEVBLK,-             ; WRITE VIRTUAL BLOCK
                                     AVAILABLE,-             ; AVAILABLE
                                     ACCESS,-                ; ACCESS FILE / FIND DIRECTORY ENTRY
                                     ACPCONTROL,-            ; ACP CONTROL FUNCTION
                                     CREATE,-                ; CREATE FILE AND/OR DIRECTORY ENTRY
                                     DEACCESS,-              ; DEACCESS FILE
                                     DELETE,-                ; DELETE FILE AND/OR DIRECTORY ENTRY
                                     MODIFY,-                ; MODIFY FILE ATTRIBUTES
                                     MOUNT-                  ; MOUNT VOLUME
                                     >
                             FUNCTAB ,-                      ;BUFFERED FUNCTIONS
                                     <NOP,-                  ; NO-OP
                                     UNLOAD,-                ; UNLOAD
                                     SEEK,-                  ; SEEK
                                     DRVCLR,-                ; DRIVE CLEAR
                                     PACKACK,-               ; PACK ACKNOWLEDGE
                                     SENSECHAR,-             ; SENSE CHARACTERISTICS
                                     SETCHAR,-               ; SET CHARACTERISTICS
                                     SENSEMODE,-             ; SENSE MODE
                                     SETMODE,-               ; SET MODE
                                     AVAILABLE,-             ; AVAILABLE
                                     ACCESS,-                ; ACCESS FILE / FIND DIRECTORY ENTRY
                                     ACPCONTROL,-            ; ACP CONTROL FUNCTION
                                     CREATE,-                ; CREATE FILE AND/OR DIRECTORY ENTRY
                                     DEACCESS,-              ; DEACCESS FILE
                                     DELETE,-                ; DELETE FILE AND/OR DIRECTORY ENTRY
                                     MODIFY,-                ; MODIFY FILE ATTRIBUTES
                                     MOUNT-                  ; MOUNT VOLUME
                                     >
                             FUNCTAB DL_ALIGN,-              ;TEST ALIGNMENT FUNCTIONS
                                     <READHEAD,-             ; READ HEADER
```

```
006C    462                     READLBLK,-              ; READ LOGICAL BLOCK
006C    463                     READPBLK,-              ; READ PHYSICAL BLOCK
006C    464                     READVBLK,-              ; READ VIRTUAL BLOCK
006C    465                     WRITECHECK,-            ; WRITE CHECK
006C    466                     WRITELBLK,-             ; WRITE LOGICAL BLOCK
006C    467                     WRITEPBLK,-             ; WRITE PHYSICAL BLOCK
006C    468                     WRITEVBLK-              ; WRITE VIRTUAL BLOCK
006C    469                     >
0078    470             FUNCTAB +ACP$READBLK,-          ;READ FUNCTIONS
0078    471                     <READHEAD,-             ; READ HEADER
0078    472                     READLBLK,-              ; READ LOGICAL BLOCK
0078    473                     READPBLK,-              ; READ PHYSICAL BLOCK
0078    474                     READVBLK-               ; READ VIRTUAL BLOCK
0078    475                     >
0084    476             FUNCTAB +ACP$WRITEBLK,-         ;WRITE FUNCTIONS
0084    477                     <WRITECHECK,-           ; WRITE CHECK
0084    478                     WRITELBLK,-             ; WRITE LOGICAL BLOCK
0084    479                     WRITEPBLK,-             ; WRITE PHYSICAL BLOCK
0084    480                     WRITEVBLK-              ; WRITE VIRTUAL BLOCK
0084    481                     >
0090    482             FUNCTAB +ACP$ACCESS,-           ;ACCESS FUNCTIONS
0090    483                     <ACCESS,-               ; ACCEESS FILE / FIND DIRECTORY ENTRY
0090    484                     CREATE-                 ; CREATE FILE AND/OR DIRECTORY ENTRY
0090    485                     >
009C    486             FUNCTAB +ACP$DEACCESS,-         ;DEACCESS FUNCTION
009C    487                     <DEACCESS-              ; DEACCESS FILE
009C    488                     >
00A8    489             FUNCTAB +ACP$MODIFY,-           ;MODIFY FUNCTIONS
00A8    490                     <ACPCONTROL,-           ; ACP CONTROL FUNCTION
00A8    491                     DELETE,-                ; DELETE FILE AND/OR DIRECTORY ENTRY
00A8    492                     MODIFY-                 ; MODIFY FILE ATTRIBUTES
00A8    493                     >
00B4    494             FUNCTAB +ACP$MOUNT,-            ;MOUNT FUNCTION
00B4    495                     <MOUNT-                 ; MOUNT VOLUME
00B4    496                     >
00C0    497             FUNCTAB +EXE$LCLDSKVALID,-      ;LOCAL DISK VALID FUNCTIONS
00C0    498                     <UNLOAD,-               ;UNLOAD VOLUME
00C0    499                      AVAILABLE,-            ;UNIT AVAILABLE
00C0    500                      PACKACK-               ;PACK ACKNOWLEDGE
00C0    501                     >
00CC    502             FUNCTAB +EXE$ZEROPARM,-         ;ZERO PARAMETER FUNCTIONS
00CC    503                     <NOP,-                  ; NO-OP
00CC    504                     UNLOAD,-                ; UNLOAD
00CC    505                     DRVCLR,-                ; DRIVE CLEAR
00CC    506                     PACKACK,-               ; PACK ACKNOWLEDGE
00CC    507                     AVAILABLE,-             ; AVAILABLE
00CC    508                     >
00D8    509             FUNCTAB +EXE$ONEPARM,-          ;ONE PARAMETER FUNCTION
00D8    510                     <SEEK-                  ; SEEK
00D8    511                     >
00E4    512             FUNCTAB +EXE$SENSEMODE,-        ;SENSE FUNCTIONS
00E4    513                     <SENSECHAR,-            ; SENSE CHARACTERISTICS
00E4    514                     SENSEMODE-              ; SENSE MODE
00E4    515                     >
00F0    516             FUNCTAB +EXE$SETCHAR,-          ;SET FUNCTIONS
00F0    517                     <SETCHAR,-              ; SET CHARACTERISTICS
00F0    518                     SETMODE-                ; SE. MODE
```

```
00F0   519                >
```

```
                     00FC   521
                     00FC   522          .SBTTL  CONTROLLER INITIALIZATION ROUTINE
                     00FC   523 ;++
                     00FC   524 ;
                     00FC   525 ; FUNCTIONAL DESCRIPTION:
                     00FC   526 ;
                     00FC   527 ;        THIS ROUTINE IS A NO-OP FOR THE RL11 BUT MUST BE INCLUDED
                     00FC   528 ;        SINCE IT IS CALLED WHEN THE RL02 IS BOOTED AS A SYSTEM DEVICE.
                     00FC   529 ;
                     00FC   530 ;        THE OPERATING SYSTEM CALLS THIS ROUTINE:
                     00FC   531 ;                - AT SYSTEM STARTUP
                     00FC   532 ;                - DURING DRIVER LOADING
                     00FC   533 ;                - DURING RECOVERY FROM POWER FAILURE
                     00FC   534 ;
                     00FC   535 ; INPUTS:
                     00FC   536 ;
                     00FC   537 ;        R4      - CSR ADDRESS (DEVICE CONTROL STATUS REGISTER)
                     00FC   538 ;        R5      - IDB ADDRESS (INTERRUPT DATA BLOCK)
                     00FC   539 ;        R6      - DDB ADDRESS (DEVICE DATA BLOCK)
                     00FC   540 ;        R8      - CRB ADDRESS (CHANNEL REQUEST BLOCK)
                     00FC   541 ;        ALL INTERRUPTS ARE LOCKED OUT
                     00FC   542 ;
                     00FC   543 ; OUTPUTS:
                     00FC   544 ;
                     00FC   545 ;        ALL REGISTERS EXCEPT R0-R3 ARE PRESERVED.
                     00FC   546 ;        CONTROL IS RETURNED TO THE CALLER.
                     00FC   547 ;
                     00FC   548 ;--
                     00FC   549
                     00FC   550 DL_RL11_INIT:                                ;CONTROLLER INITIALIZATION
                     00FC   551          ;
                     00FC   552          ; FOR MICROVAX I, ALLOCATE A PHYSICALLY CONTIGUOUS BUFFER
                     00FC   553          ; AREA FOR PERFORMING I/O.
                     00FC   554          ;
                     00FC   555          CPUDISP <<790,20$>,-
                     00FC   556                   <785,20$>,-
                     00FC   557                   <780,20$>,-
                     00FC   558                   <750,20$>,-
                     00FC   559                   <730,20$>,-
                     00FC   560                   <UV1,10$>>                 ;FOR MICROVAX I, ALLOCATE BUFFER AREA
                     012E   561                                              ;FOR ALL OTHERS, SKIP BUFFER AREA
                     012E   562
          51    14   3C   012E   563 10$:     MOVZWL  #UCB$K_DL_BUFSZ,R1      ;LOAD SIZE OF BUFFER
00000000'GF    16   0131   564          JSB     G^EXE$ALOPHYCNTG                ;ALLOCATE PHYSICALLY-CONTIGUOUS MEMORY
       05 50    E9   0137   565          BLBC    R0,20$                          ;EXIT ON ERROR
    10 A8   52    D0   013A   566          MOVL    R2,CRB$L_AUXSTRUC(R8)           ;GET BUFFER VIRTUAL ADDRESS
                  05   013E   567          RSB                                     ;RETURN TO CALLER
                       013F   568
        10 A8     D4   013F   569 20$:     CLRL    CRB$L_AUXSTRUC(R8)              ;INDICATE MEMORY ALLOCATION FAILURE
                  05   0142   570          RSB                                     ;RETURN TO CALLER
```

```
                     0143   572          .SBTTL  UNIT INITIALIZATION ROUTINE
                     0143   573
                     0143   574 ;++
                     0143   575 ;
                     0143   576 ; DL_RL0X_INIT - UNIT INITIALIZATION ROUTINE
                     0143   577 ;
                     0143   578 ; FUNCTIONAL DESCRIPTION:
                     0143   579 ;
                     0143   580 ;       THIS ROUTINE READIES THE RL01/RL02 UNITS FOR I/O OPERATIONS.
                     0143   581 ;
                     0143   582 ;       THE OPERATING SYSTEM CALLS THIS ROUTINE:
                     0143   583 ;               - AT SYSTEM STARTUP
                     0143   584 ;               - DURING DRIVER LOADING
                     0143   585 ;               - DURING RECOVERY FROM POWER FAILURE
                     0143   586 ;
                     0143   587 ; INPUTS:
                     0143   588 ;
                     0143   589 ;       R4      - CSR ADDRESS (CONTROLLER STATUS REGISTER)
                     0143   590 ;       R5      - UCB ADDRESS (UNIT CONTROL BLOCK)
                     0143   591 ;
                     0143   592 ; OUTPUTS:
                     0143   593 ;
                     0143   594 ;       THE DRIVE UNIT IS RESET, UCB FIELDS ARE INITIALIZED, AND THE
                     0143   595 ;       ROUTINE WAITS FOR ONLINE UNITS TO SPIN UP.  ALL REGISTERS
                     0143   596 ;       EXCEPT R0-R3 ARE PRESERVED.
                     0143   597 ;
                     0143   598 ;--
                     0143   599
                     0143   600 DL_RL0X_INIT:                                ;RL01/RL02 UNIT INITIALIZATION
          02   B0    0143   601          MOVW    #1@UCB$V_DL_MAPPING,-       ; DEFAULT TO ADAPTER MAPPING
      00F6 C5         0145   602                  UCB$W_DL_FLAGS(R5)          ; AND 18 BIT ADDRESSING
                      0148   603          ;
                      0148   604          ; SET CPU DEPENDENT UCB FLAGS FOR DL
                      0148   605          ;
                      0148   606          CPUDISP <<790,10$>,-
                      0148   607                   <785,10$>,-
                      0148   608                   <780,10$>,-
                      0148   609                   <750,10$>,-
                      0148   610                   <730,10$>,-
                      0148   611                   <UV1,5$>>
          01   B0    017A   612 5$:      MOVW    #1@UCB$V_DL_22BIT,-         ; FOR MICROVAX I 22 BIT
      00F6 C5         017C   613                  UCB$W_DL_FLAGS(R5)          ; ADDRESSING AND NO ADAPTER MAPPING
   53   64 A5   3C    017F   614 10$:     MOVZWL  UCB$W_STS(R5),R3            ;SAVE CURRENT UNIT STATUS
      0810 8F   AA    0183   615          BICW    #UCB$M_ONLINE!UCB$M_VALID,- ;ASSUME OFFLINE/INVALID
        64 A5         0187   616                  UCB$W_STS(R5)               ;...
                      0189   617
                      0189   618 ;
                      0189   619 ; WAIT FOR CONTROLLER (6 SECONDS MAX) IF CHANNEL IS BUSY WITH ANOTHER UNIT
                      0189   620 ;
                      0189   621
   50   24 A5   D0    0189   622          MOVL    UCB$L_CRB(R5),R0            ;GET CRB ADDRESS
29 0E A0   00   E1    018D   623          BBC     #CRB$V_BSY,CRB$B_MASK(R0),20$  ;IF CLEAR - CHANNEL NOT BUSY
                      0192   624          TIMEDWAIT TIME=#600*1000,-          ;6 SECOND WAIT LOOP
                      0192   625                  INS1=<TSTB      RL_CS(R4)>,- ;IS CONTROLLER READY
                      0192   626                  INS2=<BLSS      15$>,-      ;IF LSS - YES
                      0192   627                  DONELBL=15$                 ;LABEL TO EXIT WAIT LOOP
        3B 50   E9    01B8   628          BLBC    R0,25$                      ;TIME EXPIRED - EXIT
```

```
                                01BB   629 ;
                                01BB   630 ; GET CURRENT DRIVE STATUS AND RESET DRIVE
                                01BB   631 ;
                                01BB   632
          04 A4    0B    B0   01BB   633 20$:    MOVW    #RL_DA_M_RST!-           ;PUT RESET AND GET STATUS IN DAR
                                01BF   634                RL_DA_M_STS!RL_DA_M_MRK,RL_DA(R4)  ;...
                      51    D4   01BF   635          CLRL    R1                   ;CLEAR R1 FOR UNIT NUMBER
    51    08    08    54 A5    F0   01C1   636          INSV    UCB$W_UNIT(R5),#8,#8,R1 ;GET UNIT NUMBER
             64    04    51    A9   01C7   637          BISW3   R1,#F_GETSTATUS,RL_CS(R4)  ;EXECUTE GET STATUS FUNCTION
                   0095    30   01CB   638          BSBW    DL_WAIT              ;WAIT FOR CONTROLLER
                     64    95   01CE   639          TSTB    RL_CS(R4)            ;WAS CONTROLLER READY?
                     24    18   01D0   640          BGEQ    25$                  ;IF GEQ - NO
                                01D2   641
                                01D2   642 ;
                                01D2   643 ; CLASSIFY DRIVE TYPE
                                01D2   644 ;
                                01D2   645
          2324C001 8F    D0   01D2   646          MOVL    #^X2324C001,-
                   008C C5            01D8   647                   UCB$L_MEDIA_ID(R5)   ;SET MEDIA IDENT "DL_RL01"
       06 A4    0080 8F    B3   01DB   648          BITW    #RL_MP_M_TYP,RL_MP(R4)  ;IS DRIVE TYPE = RL02?
                     15    12   01E1   649          BNEQ    30$                  ;IF NEQ - YES
                     09    90   01E3   650          MOVB    S^#DT$_RL01,-
                     41 A5      01E5   651                   UCB$B_DEVTYPE(R5)    ;SET RL01 DEVICE TYPE
       46 A5    0100 8F    B0   01E7   652          MOVW    #256,UCB$W_CYLINDERS(R5);SET NUMBER OF RL01 CYLINDERS
    0080 C5    2800 8F    3C   01ED   653          MOVZWL  #10240,UCB$L_MAXBLOCK(R5) ;SET MAX RL01 BLOCK NUMBER
                     17    11   01F4   654          BRB     40$
                                01F6   655
                     6A    11   01F6   656 25$:    BRB     70$                  ;BRANCH TO COMMON EXIT
                                01F8   657
                     0A    90   01F8   658 30$:    MOVB    S^#DT$_RL02,-
                     41 A5      01FA   659                   UCB$B_DEVTYPE(R5)    ;SET RL02 DEVICE TYPE
       46 A5    0200 8F    B0   01FC   660          MOVW    #512,UCB$W_CYLINDERS(R5);SET NUMBER OF RL02 CYLINDERS
    0080 C5    5000 8F    3C   0202   661          MOVZWL  #20480,UCB$L_MAXBLOCK(R5) ;SET MAX RL02 BLOCK NUMBER
                008C C5    D6   0209   662          INCL    UCB$L_MEDIA_ID(R5)   ;SET MEDIA IDENT "DL RL02"
          16 53    0B    E1   020D   663 40$:    BBC     #UCB$V_VALID,R3,60$  ; Branch around wait for drive to spinup
                                0211   664                                        ;  if the drive did NOT have a VALID
                                0211   665                                        ;  volume on it before POWER failure.
                                0211   666
                                0211   667 ;
                                0211   668 ; INITIALIZE UCB FIELDS AND WAIT FOR ONLINE UNITS TO SPIN UP
                                0211   669 ;
                                0211   670
             64    01    B3   0211   671 45$:    BITW    #RL_CS_M_DRDY,RL_CS(R4) ; Is drive ready?
                     0B    12   0214   672          BNEQ    50$                  ;IF NEQ - YES
          00000000'GF    16   0216   673          JSB     G^EXE$PWRTIMCHK      ;IS MAX TIME EXCEEDED?
                F2    50   E8   021C   674          BLBS    R0,45$               ;IF LBS - NO, STILL MORE TIME NEEDED
                     06    11   021F   675          BRB     60$                  ;POWER UP TIME EXCEEDED
                                0221   676 50$:
       64 A5    0800 8F    A8   0221   677          BISW    #UCB$M_VALID,UCB$W_STS(R5) ;SET UCB STATUS VOLUME VALID
                                0227   678
                     01    E0   0227   679 60$:    BBS     #UCB$V_DL_MAPPING,-  ;ADAPTER MAPPING?
             31 00F6 C5         0229   680                   UCB$W_DL_FLAGS(R5),65$ ;IF BS YES
             51    24 A5    D0   022D   681          MOVL    UCB$L_CRB(R5),R1     ;GET CRB ADDRESS
             52    10 A1    D0   0231   682          MOVL    CRB$L_AUXSTRUC(R1),R2 ;MEMORY ALLOC FAILURE DURING CTL INIT?
                     2B    13   0235   683          BEQL    70$                  ;IF EQL YES, LEAVE OFFLINE
          00EE C5    52    D0   0237   684          MOVL    P^,UCB$A_DL_BUF_VA(R5) ;SAVE BUFFER'S VIRTUAL ADDRESS
    51    52    15    09    EF   023C   685          EXTZV   #VA$V_VPN,#VA$S_VPN,R2,R1 ;GET VIRTUAL PAGE NUMBER OF BUFFER
```

```
                50   00000000'GF   D0  0241   686          MOVL    G^MMG$GL_SPTBASE,R0      ;GET BASE ADDRESS OF SPTS
                       50   6041    D0  0248   687          MOVL    (R0)[R1],R0              ;GET THE PTE CONTENTS
           51   52   FFFFFE00 8F    CB  024C   688          BICL3   #^C<VA$M_BYTE>,R2,R1      ;GET BUFFER OFFSET (BA00-BA08)
                                     0254   689          ASSUME  PTE$S_PFN GE 13
                51   0D   09   50   F0  0254   690          INSV    R0,#9,#13,R1             ;COPY BA09-BA21
                     00F2 C5   51   D0  0259   691          MOVL    R1,UCB$A_DL_BUF_PA(R5)   ;SAVE PHYSICAL ADDRESS OF BUFFER
                       64 A5   10   A8  025E   692 65$:     BISW    #UCB$M_ONLINE,UCB$W_STS(R5)  ;SET UCB STATUS VOLUME VALID
                                    05  0262   693 70$:     RSB
```

```
                     0263   695          .SBTTL  DRIVER SPECIFIC SUBROUTINES
                     0263   696 ;
                     0263   697 ; DL_WAIT - WAIT FOR CONTROLLER READY
                     0263   698 ;
                     0263   699 ; INPUTS:
                     0263   700 ;       R4      - DEVICE CSR ADDRESS
                     0263   701 ;
                     0263   702 ; FUNCTIONAL DESCRIPTION:
                     0263   703 ;
                     0263   704 ;       THIS ROUTINE IS CALLED FROM THE DRIVER UNIT INITIALIZATION ROUTINE
                     0263   705 ;       TO WAIT UNTIL THE RL11 CONTROLLER IS READY.  TO PREVENT HANGING UP
                     0263   706 ;       AT HIGH IPL, A MAXIMUM OF 30 USEC ELAPSES BEFORE CONTROL IS
                     0263   707 ;       RETURNED TO THE CALLER.
                     0263   708 ;
                     0263   709
                     0263   710 DL_WAIT:                                      ;WAIT FOR CONTROLLER READY
7E   50   7D         0263   711          MOVQ    R0,-(SP)                    ;SAVE R0, R1
                     0266   712          DSBINT                              ;DISABLE INTERRUPTS
                     026C   713          TIMEWAIT        #3,#RL_CS_M_CRDY,RL_CS(R4),W
                     0291   714          ENBINT                              ;ENABLE INTERRUPTS
50   8E   7D         0294   715          MOVQ    (SP)+,R0                    ;RESTORE R0, R1
          05         0297   716          RSB                                 ;RETURN TO UNIT INIT OR STARTIO
```

```
                     0298   718          .SBTTL  FDT ROUTINE - TEST TRANSFER BYTE COUNT ALIGNMENT
                     0298   719
                     0298   720 ;++
                     0298   721 ;
                     0298   722 ; DL_ALIGN - FDT ROUTINE TO TEST XFER BYTE COUNT
                     0298   723 ;
                     0298   724 ; FUNCTIONAL DESCRIPTION:
                     0298   725 ;
                     0298   726 ;       THIS ROUTINE IS CALLED FROM THE FUNCTION DECISION TABLE DISPATCHER
                     0298   727 ;       TO CHECK THE BYTE COUNT PARAMETER SPECIFIED BY THE USER PROCESS
                     0298   728 ;       FOR AN EVEN NUMBER OF BYTES (WORD BOUNDARY).
                     0298   729 ;
                     0298   730 ; INPUTS:
                     0298   731 ;
                     0298   732 ;       R3      - IRP ADDRESS (I/O REQUEST PACKET)
                     0298   733 ;       R4      - PCB ADDRESS (PROCESS CONTROL BLOCK)
                     0298   734 ;       R5      - UCB ADDRESS (UNIT CONTROL BLOCK)
                     0298   735 ;       R6      - CCB ADDRESS (CHANNEL CONTROL BLOCK)
                     0298   736 ;       R7      - BIT NUMBER OF THE I/O FUNCTION CODE
                     0298   737 ;       R8      - ADDRESS OF FDT TABLE ENTRY FOR THIS ROUTINE
                     0298   738 ;       4(AP)   - ADDRESS OF FIRST FUNCTION DEPENDENT QIO PARAMETER
                     0298   739 ;
                     0298   740 ; OUTPUTS:
                     0298   741 ;
                     0298   742 ;       IF THE QIO BYTE COUNT PARAMETER IS ODD, THE I/O OPERATION IS
                     0298   743 ;       TERMINATED WITH AN ERROR. IF IT IS EVEN, CONTROL IS RETURNED
                     0298   744 ;       TO THE FDT DISPATCHER.
                     0298   745 ;
                     0298   746 ;--
                     0298   747
                     0298   748 DL_ALIGN:                                    ;CHECK BYTE COUNT AT P1(AP)
      01 04 AC   E8  0298   749          BLBS    4(AP),10$                   ;IF LBS - ODD BYTE COUNT
                 05  029C   750          RSB                                 ;EVEN - RETURN TO CALLER
50    034C 8F   3C  029D   751 10$:     MOVZWL  #SS$_IVBUFLEN,R0            ;SET BUFFER ALIGNMENT STATUS
 00000000'GF   17  02A2   752          JMP     G^EXE$ABORTIO               ;ABORT I/O
```

```
                     .SBTTL  START I/O ROUTINE

                     ;++
                     ;
                     ; DL_STARTIO - START I/O ROUTINE
                     ;
                     ; FUNCTIONAL DESCRIPTION:
                     ;
                     ;       THIS FORK PROCESS IS ENTERED FROM THE EXECUTIVE AFTER AN I/O REQUEST
                     ;       PACKET HAS BEEN DEQUEUED, AND PERFORMS THE FOLLOWING:
                     ;
                     ;               - ACTIVATES THE DISK AFTER SETTING UCB FIELDS, OBTAINING
                     ;                 UBA AND CONTROLLER RESOURCES, AND SETTING RL11 REGISTERS
                     ;
                     ;               - WAITS FOR AN INTERRUPT
                     ;
                     ;               - REGAINS CONTROL AFTER THE ISR SERVICES THE INTERRUPT, AND
                     ;                       - RE-ACTIVATES THE DISK IF THE ORIGINAL FUNCTION
                     ;                         IS NOT YET COMPLETE, OR
                     ;                       - COMPLETES THE I/O REQUEST BY RELEASING RESOURCES,
                     ;                         SETTING STATUS CODES, AND RETURNING TO THE EXECUTIVE.
                     ;
                     ; INPUTS:
                     ;
                     ;       R3              - IRP ADDRESS (I/O REQUEST PACKET)
                     ;       R5              - UCB ADDRESS (UNIT CONTROL BLOCK)
                     ;       IRP$L_MEDIA     - PARAMETER LONGWORD (LOGICAL BLOCK NUMBER)
                     ;
                     ; OUTPUTS:
                     ;
                     ;       R0      - FIRST I/O STATUS LONGWORD: STATUS CODE & BYTES XFFRED
                     ;       R1      - SECOND I/O STATUS LONGWORD: 0 FOR DISKS
                     ;
                     ;       THE I/O FUNCTION IS EXECUTED.
                     ;
                     ;       ALL REGISTERS EXCEPT R0-R4 ARE PRESERVED.
                     ;
                     ;--

                     DL_STARTIO:                            ;START I/O OPERATION

                     ;
                     ;       COMPUTE PHYSICAL MEDIA ADDRESS
                     ;
                     ;               LBN = LBN * (SECTORS/BLOCK)
                     ;               LBN/(SECTORS/TRACK) = D + SECTOR
                     ;               D/(TRACKS/CYLINDER) = CYLINDER + TRACK
                     ;


                     ;
                     ;       PREPROCESS UCB FIELDS
                     ;

                     PREPROCESS:                            ;
  38 A3   D0         MOVL    IRP$L_MEDIA(R3),-           ; Copy given MEDIA address (logical)
0OBC C5              UCB$L_MEDIA(R5)             ;  to the UCB.
    08   E0         BBS     #IRP$V_PHYSIO,-             ;IF SET - PHYSICAL I/O
```

```
                    26 2A A3        02B0   811                      IRP$W_STS(R3),10$
          50    00BC C5    02   C5  02B3   812              MULL3   #2,UCB$L_MEDIA(R5),R0   ;SCALE LBN IN R0
                  52    44 A5   9A  02B9   813              MOVZBL  UCB$B_SECTORS(R5),R2    ;GET NUMBER OF SECTORS PER TRACK
                             51   D4  02BD   814              CLRL    R1                      ;CLEAR HIGH PART OF DIVIDEND
                00BC C5    50    50   52   7B  02BF   815              EDIV    R2,R0,R0,UCB$L_MEDIA(R5);CALCULATE SECTOR NUMBER AND STORE
                      52    45 A5   9A  02C6   816              MOVZBL  UCB$B_TRACKS(R5),R2     ;GET NUMBER OF TRACKS PER CYLINDER
          51    50    50    52   7B  02CA   817              EDIV    R2,R0,R0,R1             ;CALCULATE TRACK AND CYLINDER
                00BD C5    51   90  02CF   818              MOVB    R1,UCB$L_MEDIA+1(R5)    ;STORE TRACK NUMBER
                00BE C5    50   B0  02D4   819              MOVW    R0,UCB$L_MEDIA+2(R5)    ;STORE CYLINDER NUMBER
                                    02D9   820 10$:
                      0081 C5   90  02D9   821              MOVB    UCB$B_ERTMAX(R5),-      ;INITIALIZE ERROR RETRY COUNT
                      0080 C5       02DD   822                      UCB$B_ERTCNT(R5)        ;...
                00C0 C5    7E A5   AE  02E0   823              MNEGW   UCB$W_BCNT(R5),UCB$W_BCR(R5)  ;INIT NEG BYTES LEFT TO XFER
                      00D6 C5   B4  02E6   824              CLRW    UCB$W_DL_DPN(R5)        ;CLEAR DATA PATH NO. FOR USE AS-
                                    02EA   825                                              ;UBA RESOURCE ALLOCATION FLAG
                      00E4 C5   94  02EA   826              CLRB    UCB$B_DL_DPPE(R5)       ;CLEAR DATAPATH PURGE ERROR REGISTER
                009A C5    20 A3   B0  02EE   827              MOVW    IRP$W_FUNC(R3),UCB$W_FUNC(R5)  ;SAVE FUNCTION CODE
                             00   EF  02F4   828              EXTZV   #IRP$V_FCODE,-          ;EXTRACT I/O FUNCTION CODE
                51    20 A3    06       02F6   829                      #IRP$S_FCODE,IRP$W_FUNC(R3),R1  ;...
                   0092 C5    51   90  02FA   830              MOVB    R1,UCB$B_FEX(R5)        ;STORE FUNCTION DISPATCH INDEX
                      51    02   91  02FF   831              CMPB    #IO$_SEEK,R1            ;SEEK FUNCTION?
                             06   12  0302   832              BNEQ    20$                     ;IF NEQ - NO
                       38 A3   B0  0304   833              MOVW    IRP$L_MEDIA(R3),-       ;STORE CYLINDER ADDRESS
                      00BE C5       0307   834                      UCB$W_DC(R5)            ;...
                                    030A   835 20$:
                             02   AA  030A   836              BICW    #UCB$M_DIAGBUF,-
                                                                    UCB$W_DEVSTS(R5)        ;CLR DIAGNOSTIC BUFFER PRESENT
                         68 A5       030C   837
                             07   E1  030E   838              BBC     #IRP$V_DIAGBUF,-        ;IF CLR - NO DIAG BUFFER
                       04 2A A3       0310   839                      IRP$W_STS(R3),FDISPATCH ;...
                    68 A5    02   A8  0313   840              BISW    #UCB$M_DIAGBUF,UCB$W_DEVSTS(R5)  ;SET DIAG BUFFER PRESENT
                                    0317   841
                                    0317   842 ;
                                    0317   843 ;       CENTRAL FUNCTION DISPATCH
                                    0317   844 ;
                                    0317   845
                                    0317   846 FDISPATCH:                                    ;FUNCTION DISPATCH
              53    58 A5    D0   0317   847              MOVL    UCB$L_IRP(R5),R3        ;GET IRP ADDRESS
                         08   E0  031B   848              BBS     #IRP$V_PHYSIO,-         ;IF SET - PHYSICAL I/O FUNCTION
                 0D 2A A3       031D   849                      IRP$W_STS(R3),10$       ;...
                         0B   E0  0320   850              BBS     #UCB$V_VALID,-          ;IF SET - V  ME SOFTWARE VALID
                08 64 A5        0322   851                      UCB$W_STS(R5),10$       ;...
          50     0254 8F    3C   0325   852              MOVZWL  #SS$_VOLINV,R0          ;SET VOLUME INVALID STATUS
                   05AE    31   032A   853              BRW     RESETXFR                ;RESET BYTE COUNT AND EXIT
                 00C9 C5    94   032D   854 10$:         CLRB    UCB$B_DL_DCHEK(R5)      ;CLEAR DATA CHECK IN PROGRESS
              53    0092 C5    9A   0331   855              MOVZBL  UCB$B_FEX(R5),R3        ;GET FUNCTION DISPATCH INDEX
                                    0336   856              CASE    R3,<-                   ;DISPATCH TO FUNCTION HANDLING ROUTINE
                                    0336   857                      UNLOAD,-                ; UNLOAD
                                    0336   858                      SEEK,-                  ; SEEK
                                    0336   859                      NOP,-                   ; RECALIBRATE (unsupported)
                                    0336   860                      DRVCLR,-                ; DRVCLR
                                    0336   861                      NOP,-                   ; RELEASE PORT (unsupported)
                                    0336   862                      NOP,-                   ; OFFSET HEADS (unsupported)
                                    0336   863                      NOP,-                   ; RETURN TO CENTER (unsupported)
                                    0336   864                      PACKACK,-               ; PACK ACKNOWLEDGE
                                    0336   865                      NOP,-                   ; SEARCH (unsupported)
                                    0336   866                      WRITECHECK,-            ; WRITE CHECK
                                    0336   867                      WRITEDATA,-             ; WRITE DATA
```

```
                     0336   868                 READDATA,-                      ; READ DATA
                     0336   869                 NOP,-                           ; WRITE HEADER (unsupported)
                     0336   870                 READHEAD,-                      ; READ HEADER
                     0336   871                 NOP,-                           ; place holder
                     0336   872                 NOP,-                           ; place holder
                     0336   873                 AVAILABLE-                      ; AVAILABLE
                     0336   874                 >,LIMIT=#CDF_UNLOAD             ;
                     035C   875
                     035C   876 NOP:                                            ;NO-OP
                     035C   877 SEEK:                                           ;SEEK
                     035C   878 DRVCLR:                                         ;DRIVE CLEAR (GET STATUS & RESET)
                     035C   879 DO_FUNCTION:
                     035C   880         EXFUNCL RETRYERR                       ;EXECUTE FUNCTION - RETRY IF FAILURE
          27   11    0360   881         BRB     NORMAL                         ;SUCCESSFUL - EXIT WITH NORMAL STATUS
                     0362   882
                     0362   883 PACKACK:                                       ;PACK ACKNOWLEDGE (GET STATUS & RESET)
64 A5     0800 8F    A8 0362   884         BISW    #UCB$M_VALID, -                ;Set software volume valid bit.
                     0368   885                 UCB$W_STS(R5)
          F2   11    0368   886         BRB     DO_FUNCTION                    ;Then go do hardware function.
                     036A   887
                     036A   888 UNLOAD:                                         ;UNLOAD
                     036A   889 AVAILABLE:                                      ;AVAILABLE
64 A5     0800 8F    AA 036A   890         BICW    #UCB$M_VALID, -                ;Clear software volume valid bit.
                     0370   891                 UCB$W_STS(R5)                  ;and go complete operation without
          17   11    0370   892         BRB     NORMAL                         ;any hardware interaction.
                     0372   893
                     0372   894 WRITECHECK:                                     ;WRITE CHECK
                     0372   895 READHEAD:                                      ;READ HEADER
          4000 8F    AA 0372   896         BICW    #IO$M_DATACHECK,-              ;CLEAR DATA CHECK REQUEST-
          009A C5    0376   897                 UCB$W_FUNC(R5)                 ;TO PREVENT EXTRA WRITE CHECK
                     0379   898
                     0379   899 WRITEDATA:                                     ;WRITE DATA
                     0379   900 READDATA:                                      ;READ DATA
                     0379   901         EXFUNCL RETRYERR,F_SEEK                ;EXECUTE EXPLICIT SEEK - RETRY IF FAIL
                     0380   902
53        0092 C5    9A 0380   903         MOVZBL  UCB$B_FEX(R5),R3               ;GET FUNCTION DISPATCH INDEX
                     0385   904         EXFUNCL RETRYERR                       ;EXECUTE TRANSFER FUNCTION
                     0389   905
                     0389   906 ;
                     0389   907 ;       OPERATON COMPLETION
                     0389   908 ;
                     0389   909
                     0389   910 NORMAL:                                        ;SUCCESSFUL OPERATION COMPLETE
     50    01    3C 0389   911         MOVZWL  #SS$_NORMAL,R0                 ;SET NORMAL COMPLETION STATUS
        0057    31 038C   912         BRW     FUNCXT                         ;FUNCTION EXIT
                     038F   913
                     038F   914 RETRYERR:                                       ;RETRIABLE ERROR
     0080 C5    97 038F   915         DECB    UCB$B_ERTCNT(R5)               ;ANY RETRIES LEFT?
          03    13 0393   916         BEQL    FATALERR                       ;IF EQL - NO
        FF7F    31 0395   917         BRW     FDISPATCH                      ;RETRY FUNCTION
                     0398   918
                     0398   919 FATALERR:                                      ;UNRECOVERABLE ERROR
50        0254 8F    3C 0398   920         MOVZWL  #SS$_VOLINV,R0                 ;ASSUME VOLUME INVALID STATUS
          09   E0    039D   921         BBS     #RL_MP_V_VC,-                  ;IF SET - VOLUME INVALID
   43 00D4 C5         039F   922                 UCB$W_DL_MP(R5),FUNCXT         ;...
                     03A3   923
50        025C 8F    3C 03A3   924         MOVZWL  #SS$_WRITLCK,R0                ;ASSUME WRITE LOCK ERROR STATUS
```

```
                 0D   E1  03A8   925          BBC     #RL_MP_V_WL,-                ;IF CLR - VOLUME NOT WRITE LOCKED
       06 00D4   C5       03AA   926                  UCB$W_DL_MP(R5),5$           ;...
                 0A   E0  03AE   927          BBS     #RL_MP_V_WGE,-               ;IF SET - WRITE GATE ERROR
       32 00D4   C5       03B0   928                  UCB$W_DL_MP(R5),FUNCXT       ;IF WL & WGE SET - WRITE LOCK ERROR
                          03B4   929
   50    005C   8F   3C  03B4   930 5$:     MOVZWL  #SS$_DATACHECK,R0            ;ASSUME DATA CHECK ERROR STATUS
         00C9   C5   95  03B9   931          TSTB    UCB$B_DL_DCHEK(R5)           ;WRITE CHECK IN PROGRESS?
                 0C   13  03BD   932          BEQL    10$                          ;IF EQL - NO
                 0A   E0  03BF   933          BBS     #RL_CS_V_OPI,-               ;IF SET - NOT WRITE CHECK ERROR
       06 00CE   C5       03C1   934                  UCB$W_DL_CS(R5),10$          ;...
                 0B   E0  03C5   935          BBS     #RL_CS_V_CRC,-               ;IF SET - WRITE CHECK ERROR
       1B 00CE   C5       03C7   936                  UCB$W_DL_CS(R5),FUNCXT       ;...
                          03CB   937
   50    01F4   8F   3C  03CB   938 10$:     MOVZWL  #SS$_PARITY,R0               ;ASSUME PARITY ERROR STATUS
                 0B   E0  03D0   939          BBS     #RL_CS_V_CRC,-               ;IF SET - CRC ERROR
       10 00CE   C5       03D2   940                  UCB$W_DL_CS(R5),FUNCXT       ;OR DATAPATH PURGE ERROR
                          03D6   941
   50    008C   8F   3C  03D6   942 20$:     MOVZWL  #SS$_DRVERR,R0               ;ASSUME DRIVE ERROR STATUS
                 0E   E0  03DB   943          BBS     #RL_CS_V_DE,-                ;IF SET - DRIVE ERROR
       05 00CE   C5       03DD   944                  UCB$W_DL_CS(R5),FUNCXT       ;...
                          03E1   945
   50    0054   8F   3C  03E1   946          MOVZWL  #SS$_CTRLERR,R0              ;ASSUME CONTROLLER ERROR STATUS
                          03E6   947
                          03E6   948 FUNCXT:                                       ;FUNCTION EXIT
                 50   DD  03E6   949          PUSHL   R0                           ;SAVE FINAL REQUEST STATUS
    00000000'GF   16       03E8   950          JSB     G^IOC$DIAGBUFILL             ;FILL DIAGNOSTIC BUFFER IF PRESENT
 0092 C5         0A   91  03EE   951          CMPB    #CDF_WRITECHECK,UCB$B_FEX(R5)  ;DRIVE RELATED FUNCTION?
                 2D   1A  03F3   952          BGTRU   10$                          ;IF GTRU - YES
 0092 C5         11   91  03F5   953          CMPB    #CDF_AVAILABLE,UCB$B_FEX(R5)   ;DRIVE RELATED FUNCTION?
                 26   13  03FA   954          BEQL    10$                          ;IF EQL - YES
      53   58    A5   D0  03FC   955          MOVL    UCB$L_IRP(R5),R3             ;RETRIEVE ADDRESS OF IRP
           00C0  C5   A1  0400   956          ADDW3   UCB$W_BCR(R5),-              ;CALCULATE BYTES TRANSFERRED
   02 AE     32   A3       0404   957                  IRP$W_BCNT(R3),2(SP)         ;...
           00D6  C5   B5  0408   958          TSTW    UCB$W_DL_DPN(R5)             ;ARE UBA RESOURCES ALLOCATED?
                 1A   13  040C   959          BEQL    20$                          ;IF EQL - NO
                 01   E1  040E   960          BBC     #UCB$V_DL_MAPPING,-          ;ADAPTER MAPPING?
      0E 00F6   C5       0410   961                  UCB$W_DL_FLAGS(R5),10$       ;IF BC NO
                          0414   962          RELDPR                               ;RELEASE DATA PATH
                          041A   963          RELMPR                               ;RELEASE MAP REGISTERS
                 06   11  0420   964          BRB     20$                          ;JOIN COMMON CODE
           00D8  C5   D0  0422   965 10$:     MOVL    UCB$L_DL_SVAPTE(R5),-        ;RESTORE ORIGINAL SVAPTE
             78  A5       0426   966                  UCB$L_SVAPTE(R5)             ;
                          0428   967 20$:     RELCHAN                              ;RELEASE CHANNEL IF OWNED
                          042E   968
                 51   D4  042E   969          CLRL    R1                           ;CLEAR SECOND STATUS LONGWORD
                 50 8ED0  0430   970          POPL    R0                           ;RETRIEVE FINAL REQUEST STATUS
                          0433   971          REQCOM                               ;COMPLETE REQUEST
```

```
0439   973 ;
0439   974 ; FEXL - RL11 HARDWARE FUNCTION EXECUTION
0439   975 ;
0439   976 ; THIS ROUTINE IS CALLED VIA A BSB WITH A BYTE IMMEDIATELY FOLLOWING THAT
0439   977 ; SPECIFIES THE ADDRESS OF AN ERROR ROUTINE. ALL DATA IS ASSUMED TO HAVE BEEN
0439   978 ; SET UP IN THE UCB BEFORE THE CALL. THE APPROPRIATE PARAMETERS ARE LOADED
0439   979 ; INTO DEVICE REGISTERS AND THE FUNCTION IS INITIATED. THE RETURN ADDRESS
0439   980 ; IS STORED IN THE UCB AND A WAITFOR INTERRUPT IS EXECUTED. WHEN THE
0439   981 ; INTERRUPT OCCURS, CONTROL IS RETURNED TO THE CALLER.
0439   982 ;
0439   983 ; INPUTS:
0439   984 ;
0439   985 ;       R3 = FUNCTION TABLE DISPATCH INDEX
0439   986 ;       R5 = DEVICE UNIT UCB ADDRESS
0439   987 ;
0439   988 ;       00(SP) = RETURN ADDRESS OF CALLER
0439   989 ;       04(SP) = RETURN ADDRESS OF CALLER'S CALLER
0439   990 ;
0439   991 ;       IMMEDIATELY FOLLOWING INLINE AT THE CALL SITE IS A BYTE WHICH CONTAINS
0439   992 ;       A BRANCH DESTINATION TO AN ERROR RETRY ROUTINE.
0439   993 ;
0439   994 ; OUTPUTS:
0439   995 ;
0439   996 ;       THERE ARE FOUR EXITS FROM THIS ROUTINE:
0439   997 ;
0439   998 ;       1. SPECIAL CONDITION - THIS EXIT IS TAKEN IF A POWER FAILURE OCCURS
0439   999 ;               OR THE OPERATION TIMES OUT. IT IS A JUMP TO THE APPROPRIATE
0439  1000 ;               ERROR ROUTINE.
0439  1001 ;
0439  1002 ;       2. FATAL ERROR - THIS EXIT IS TAKEN IF A FATAL CONTROLLER OR DRIVE
0439  1003 ;               ERROR OCCURS OR IF ANY ERROR OCCURS AND ERROR RETRY IS EITHER
0439  1004 ;               INHIBITED OR EXHAUSTED. IT IS A JUMP TO THE FATAL ERROR EXIT
0439  1005 ;               ROUTINE.
0439  1006 ;
0439  1007 ;       3. RETRIABLE ERROR - THIS EXIT IS TAKEN IF A RETRIABLE CONTROLLER
0439  1008 ;               OR DRIVE ERROR OCCURS AND ERROR RETRY IS NEITHER INHIBITED
0439  1009 ;               NOR EXHAUSTED. IT CONSISTS OF TAKING THE ERROR BRANCH EXIT
0439  1010 ;               SPECIFIED AT THE CALL SITE.
0439  1011 ;
0439  1012 ;       4. SUCCESSFUL OPERATION - THIS EXIT IS TAKEN IF NO ERRORS OCCUR
0439  1013 ;               DURING THE OPERATION. IT CONSISTS OF A RETURN INLINE.
0439  1014 ;
0439  1015 ;       IN ALL CASES IF AN ERROR OCCURS, AN ATTEMPT IS MADE TO LOG THE ERROR.
0439  1016 ;
0439  1017 ;       IN ALL CASES FINAL DEVICE REGISTERS ARE RETURNED VIA THE UCB.
0439  1018 ;
0439  1019 ;       UCB$W_BCR(R5) = NEGATIVE BYTES REMAINING TO TRANSFER
```

```
                     0439  1021 FEXL:                                          ;FUNCTION EXECUTOR
     009C C5 8ED0    0439  1022          POPL    UCB$L_DPC(R5)                  ;SAVE DRIVER PC VALUE
0093 C5    53   90    043E  1023          MOVB    R3,UCB$B_CEX(R5)               ;SAVE CASE INDEX
  50   24 A5   D0    0443  1024          MOVL    UCB$L_CRB(R5),R0               ;GET ADDRESS OF PRIMARY CRB
  51   2C A0   D0    0447  1025          MOVL    CRB$L_INTD+VEC$L_IDB(R0),R1  ;GET ADDRESS OF IDB
  04 A1    55   D1    044B  1026          CMPL    R5,IDB$L_OWNER(R1)             ;DOES THIS PROCESS OWN CHANNEL?
           05   12    044F  1027          BNEQ    10$                            ;IF NEQ - NO
      54   61   D0    0451  1028          MOVL    IDB$L_CSR(R1),R4               ;SET ASSIGNED CHANNEL CSR ADDRESS
           06   11    0454  1029          BRB     20$                            ;
                      0456  1030 10$:     REQPCHAN                               ;REQUEST CHANNEL (RETURNS R4 = CSR ADR)
                      045C  1031
                      045C  1032 20$:     CASE    R3,<-                          ;DISPATCH TO PROPER FUNCTION ROUTINE
                      045C  1033                  IMMED,-                        ;NO OPERATION
                      045C  1034                  IMMED,-                        ;UNLOAD VOLUME (NOP)
                      045C  1035                  POSIT,-                        ;SEEK CYLINDER
                      045C  1036                  IMMED,-                        ;RECALIBRATE (NOP)
                      045C  1037                  DRCLR,-                        ;DRIVE CLEAR (GET STATUS & RESET)
                      045C  1038                  IMMED,-                        ;RELEASE DRIVE (NOP)
                      045C  1039                  IMMED,-                        ;OFFSET HEADS (NOP)
                      045C  1040                  IMMED,-                        ;RETURN TO CENTERLINE (NOP)
                      045C  1041                  DRCLR,-                        ;PACK ACKNOWLEDGE
                      045C  1042                  IMMED,-                        ;SEARCH (NOP)
                      045C  1043                  >                              ;
          01A3   31   0474  1044          BRW     XFER                           ;TRANSFER FUNCTION
```

```
                     0477  1046 ;
                     0477  1047 ; IMMEDIATE FUNCTION EXECUTION
                     0477  1048 ;
                     0477  1049 ;       FUNCTIONS INCLUDE:
                     0477  1050 ;
                     0477  1051 ;               NO OPERATION,
                     0477  1052 ;               DRIVE CLEAR, AND
                     0477  1053 ;               PACK ACKNOWLEDGE
                     0477  1054 ;
                     0477  1055 ; INPUTS:
                     0477  1056 ;       R3      - CASE INDEX
                     0477  1057 ;       R4      - CSR ADDRESS
                     0477  1058 ;       R5      - UCB ADDRESS
                     0477  1059 ;
                     0477  1060 ; FUNCTIONAL DESRIPTION:
                     0477  1061 ;
                     0477  1062 ; INTERRUPTS ARE LOCKED OUT, THE APPROPRIATE FUNCTION IS INITIATED WITH
                     0477  1063 ; INTERRUPT ENABLE, AND A WAITFOR INTERRUPT AND KEEP CHANNEL IS EXECUTED.
                     0477  1064 ;
                     0477  1065
                     0477  1066 DRCLR:                                          ;DRIVE CLEAR
        04 A4   08   A8  0477  1067         BISW    #RL_DA_M_STS!-              ;SET GETSTATUS,RESET,AND MARK IN DAR
                     047B  1068                 RL_DA_M_RST!RL_DA_M_MRK,RL_DA(R4)  ;...
                     047B  1069
                     047B  1070 IMMED:                                          ;IMMEDIATE FUNCTION EXECUTION
                     047B  1071         CKPWR                                   ;DISABLE INTERRUPTS, CHECK POWER,-
                     0494  1072                                                 ;AND PUT UNIT NUMBER IN R2<9:8>
    64  FB9E CF43   52   A9  0494  1073         BISW3   R2,FTAB[R3],RL_CS(R4)   ;MERGE UNIT WITH FNTN AND EXECUTE
                     049B  1074         WFIKPCH RETREG,#2                       ;WAITFOR INTERRUPT
                     04A5  1075         IOFORK                                  ;RETURN FROM ISR-
                     04AB  1076                                                 ;CREATE FORK PROCESS (&JSB BACK TO ISR)
              0371   31   04AB  1077         BRW     RETREG                      ;
```

```
                     04AE  1079 ;
                     04AE  1080 ; POSITIONING FUNCTION EXECUTION
                     04AE  1081 ;
                     04AE  1082 ;       FUNCTIONS INCLUDE:
                     04AE  1083 ;
                     04AE  1084 ;               SEEK CYLINDER
                     04AE  1085 ;
                     04AE  1086 ; INPUTS:
                     04AE  1087 ;       R3      - CASE INDEX
                     04AE  1088 ;       R4      - DEVICE CSR ADDRESS
                     04AE  1089 ;       R5      - UCB ADDRESS
                     04AE  1090 ;
                     04AE  1091 ; FUNCTIONAL DESRIPTION:
                     04AE  1092 ;
                     04AE  1093 ; THE CYLINDER DIFFERENCE WORD IS CALCULATED AND LOADED INTO THE DISK
                     04AE  1094 ; ADDRESS REGISTER, INTERRUPTS ARE LOCKED OUT, AND THE SEEK FUNCTION
                     04AE  1095 ; IS INITIATED WITHOUT INTERRUPT ENABLE. THE CONTROLLER IS THEN POLLED
                     04AE  1096 ; FOR READY, AND DEVICE INTERRUPTS ARE ENABLED.
                     04AE  1097 ;
                     04AE  1098 ; SINCE THE RL01/RL02 DO NOT ISSUE AN INTERRUPT UPON COMPLETION OF A
                     04AE  1099 ; SEEK, OVERLAPPED SEEKS ARE NOT ATTEMPTED, AND ONE OF THE FOLLOWING IS
                     04AE  1100 ; PERFORMED.
                     04AE  1101 ;
                     04AE  1102 ;       IF ONLY A SEEK FUNCTION IS BEING REQUESTED, A DUMMY READ HEADER
                     04AE  1103 ;       FUNCTION IS ISSUED AND A WAITFOR INTERRUPT IS INITIATED.
                     04AE  1104 ;       THE READ HEADER IS USED TO SIGNAL THE END OF THE SEEK, SINCE IT
                     04AE  1105 ;       WILL ISSUE AN INTERRUPT SHORTLY (315 USEC AVG) AFTER THE SEEK IS
                     04AE  1106 ;       COMPLETE. IT WILL ALSO SENSE FOR A TIMEOUT DURING THE SEEK.
                     04AE  1107 ;
                     04AE  1108 ;       IF THE SEEK IS ASSOCIATED WITH A DATA TRANSFER REQUEST (RL01/RL02
                     04AE  1109 ;       TRANSFER FUNCTIONS REQUIRE EXPLICIT SEEKS), THE PROGRAM KEEPS THE
                     04AE  1110 ;       CHANNEL AND RETURNS TO FDISPATCH TO ISSUE THE TRANSFER REQUEST
                     04AE  1111 ;       WHILE THE SEEK IS STILL IN PROGREES. WHEN THE SEEK COMPLETES, THE
                     04AE  1112 ;       RL11 CONTROLLER WILL BEGIN THE TRANSFER.
                     04AE  1113 ;
                     04AE  1114
                     04AE  1115 POSIT:                                       ;POSITIONING FUNCTION
                     04AE  1116 ;
                     04AE  1117 ; OBTAIN CURRENT DISK ADDRESS
                     04AE  1118 ;
                     04AE  1119 ; IF THERE HAS NOT BEEN A PREVIOUS TRANSFER DURING THIS REQUEST,
                     04AE  1120 ; A READ HEADER IS EXECUTED TO DETERMINE THE CURRENT DISK ADDRESS.
                     04AE  1121 ;
          00D6 C5    B5  04AE  1122         TSTW    UCB$W_DL_DPN(R5)         ;WAS THERE A PREVIOUS TRANSFER?
               09    13  04B2  1123         BEQL    10$                      ;IF EQL - NO, READ HEADER
51   00D2 C5   3F    AB  04B4  1124         BICW3   #^O77,UCB$W_DL_DA(R5),R1  ;PUT CURRENT CYL & SURFACE IN R1
               00B5  31  04BA  1125         BRW     60$                      ;CALCULATE DIFFERENCE WORD
                     04BD  1126
          53   08    9A  04BD  1127 10$:     MOVZBL  #8,R3                    ;SET READ HEADER RETRY COUNT IN R3
                     04C0  1128 20$:     CKPWR                            ;DISABLE INTERRUPTS, CHECK POWER,-
                     04D9  1129                                           ;AND PUT UNIT NUMBER IN R2<9:8>
     0048 8F   52    A9  04D9  1130         BISW3   R2,#6F_READHEAD!RL_CS_M_IE,-  ;EXECUTE READ HEADER
               64        04DE  1131                 RL_CS(R4)                ;...
                     04DF  1132         WFIKPCH 40$,#2                   ;WAIT FOR INTERRUPT OR TIMEOUT
                     04E9  1133         IOFORK                           ;CREATE FORK PROCESS
00CE C5   8000 8F   B3  04EF  1134         BITW    #RL_CS_M_CE,UCB$W_DL_CS(R5)  ;ANY ERRORS?
               74   13  04F6  1135         BEQL    50$                      ;IF EQL - NO
```

```
                                                     H 7
                     53   97   04F8  1136          DECB    R3                              ;DECREMENT READ HEADER RETRY COUNT
                     C4   12   04FA  1137          BNEQ    20$                             ;IF NEQ - RETRY READ HEADER
                               04FC  1138                                                  ;IF EQL - READ HEADER RETRY EXHAUSTED -
                               04FC  1139                                                  ;TRY PREVIOUS TRACK
                0081 8F   B0   04FC  1140          MOVW    #^O200!RL_DA_M_MRK,-            ;LOAD REVERSE SEEK DIFFERENCE WORD
                  04 A4        0500  1141                  RL_DA(R4)                       ;...
                               0502  1142          CKPWR                                   ;DISABLE INTERRUPTS, CHECK POWER,-
                               051B  1143                                                  ;AND PUT UNIT NUMBER IN R2<9:8>
           0046 8F    52   A9   051B  1144          BISW3   R2,#F_SEEK!RL_CS_M_IE,-         ;EXECUTE REVERSE SEEK
                     64         0520  1145                  RL_CS(R4)                       ;...
                               0521  1146          WFIKPCH 40$,#2                          ;WAIT FOR SEEK TO BEGIN (INTERRUPT)
                               052B  1147          IOFORK                                  ;CREATE FORK PROCESS
                               0531  1148          CKPWR                                   ;DISABLE INTERRUPTS, CHECK POWER,-
                               054A  1149                                                  ;AND PUT UNIT NUMBER IN R2<9:8>
           0048 8F    52   A9   054A  1150          BISW3   R2,#F_READHEAD!RL_CS_M_IE,-  ;TRY READ HEADER ON NEW TRACK
                     64         054F  1151                  RL_CS(R4)                       ;...
                               0550  1152          WFIKPCH 40$,#2                          ;WAITFOR INTERRUPT OR TIMEOUT
                               055A  1153          IOFORK                                  ;CREATE FORK PROCESS
      00CE C5    8000 8F    B3   0560  1154          BITW    #RL_CS_M_CE,UCB$W_DL_CS(R5)  ;READ HEADER ERROR?
                     03   13   0567  1155          BEQL    50$                             ;IF EQL - NO
                               0569  1156 40$:                                             ;CAN NOT READ CURRENT DISK ADDRESS
                               0569  1157 ;         CLRB    UCB$B_ERTCNT(R5)                ;CLEAR RETRY COUNT
                          02B3 31   0569  1158          BRW     RETREG                          ;
                               056C  1159 50$:                                             ;FOUND CURRENT DISK ADDRESS
      51   00D4 C5    3F   AB   056C  1160          BICW3   #^O77,UCB$W_DL_MP(R5),R1  ;PUT CURRENT CYL & SURFACE IN R1
                               0572  1161
                               0572  1162 ;
                               0572  1163 ; CALCULATE CYLINDER DIFFERENCE WORD
                               0572  1164 ;
                               0572  1165
                     50   D4   0572  1166 60$:     CLRL    R0                              ;CLEAR R0 FOR DESIRED ADDRESS
50   01   06   00BD C5   F0   0574  1167          INSV    UCB$W_DA+1(R5),#6,#1,R0         ;INSERT DESIRED SURFACE IN R0<6>
50   09   07   00BE C5   F0   057B  1168          INSV    UCB$W_DC(R5),#7,#9,R0           ;INSERT DESIRED CYLINDER IN R0<15:7>
               51   50   B1   0582  1169          CMPW    R0,R1                           ;IS A SEEK NEEDED?
                     52   13   0585  1170          BEQL    80$                             ;IF EQL - NO
               51   007F 8F   AA   0587  1171          BICW    #^O177,R1                       ;REMOVE SURFACE BIT
               50   007F 8F   AA   058C  1172          BICW    #^O177,R0                       ;REMOVE SURFACE BIT
               51   50   A2   0591  1173          SUBW    R0,R1                           ;SUBTRACT DESIRED FROM ACTUAL
                     08   13   0594  1174          BEQL    70$                             ;IF EQL - ONLY CHANGE SURFACE
                     06   1E   0596  1175          BCC     70$                             ;IF CC - ACTUAL>=DESIRED
               51   51   AE   0598  1176          MNEGW   R1,R1                           ;ACTUAL<DESIRED, MAKE POSITIVE DIFF
               51   04   A8   059B  1177          BISW    #4,R1                           ;SET SIGN FOR MOVE TO CENTER OF DISK
51   01   04   00BD C5   F0   059E  1178 70$:     INSV    UCB$W_DA+1(R5),#4,#1,R1         ;INSERT SURFACE BIT
      04 A4   51   01   A9   05A5  1179          BISW3   #RL_DA_M_MRK,R1,RL_DA(R4)  ;SET MARKER AND LOAD DIFFERENCE WORD
                               05AA  1180
                               05AA  1181 ;
                               05AA  1182 ; EXECUTE SEEK
                               05AA  1183 ;
                               05AA  1184
                               05AA  1185          CKPWR                                   ;DISABLE INTERRUPTS, CHECK POWER,-
                               05C3  1186                                                  ;AND PUT UNIT NUMBER IN R2<9:8>
           0046 8F    52   A9   05C3  1187          BISW3   R2,#F_SEEK!RL_CS_M_IE,-         ;EXECUTE SEEK FUNCTION
                     64         05C8  1188                  RL_CS(R4)                       ;...
                               05C9  1189          WFIKPCH 40$,#2                          ;WAIT FOR SEEK TO BEGIN (INTERRUPT)
                               05D3  1190          IOFORK                                  ;CREATE FORK PROCESS
           0092 C5    02   91   05D9  1191 80$:     CMPB    #IO$_SEEK,UCB$B_FEX(R5)         ;IS SEEK ASSOCIATED WITH A TRANSFER?
                     08   13   05DE  1192          BEQL    90$                             ;IF EQL - NO, SEEK ONLY
```

```
                     05E0  1193
                     05E0  1194 ;
                     05E0  1195 ; RETURN FOR SEEK ASSOCIATED WITH A TRANSFER REQUEST
                     05E0  1196 ;
                     05E0  1197
        009C C5  D6  05E0  1198          INCL    UCB$L_DPC(R5)           ;ADJUST TO CORRECT RETURN ADDRESS
        009C D5  17  05E4  1199          JMP     @UCB$L_DPC(R5)          ;RETURN TO DRIVER FOR TRANSFER
                     05E8  1200 ;
                     05E8  1201 ; RETURN FOR SEEK ONLY REQUEST
                     05E8  1202 ;
                     05E8  1203
                     05E8  1204 90$:  CKPWR                           ;DISABLE INTERRUPTS, CHECK POWER,-
                     0601  1205                                       ;AND PUT UNIT NUMBER IN R2<9:8>
0048 8F    52  A9    0601  1206          BISW3   R2,#F_READHEAD!RL_CS_M_IE,-  ;EXECUTE DUMMY READ HEADER
              64         0606  1207                  RL_CS(R4)               ;...
                     0607  1208          WFIKPCH RETREG,#2               ;WAIT FOR SEEK TO COMPLETE (INTERRUPT)
                     0611  1209          IOFORK                          ;CREATE FORK PROCESS
            0205  31     0617  1210          BRW     RETREG                  ;
```

```
                     061A  1212
                     061A  1213 ;
                     061A  1214 ; TRANSFER FUNCTION EXECUTION
                     061A  1215 ;
                     061A  1216 ;       FUNCTIONS INCLUDE:
                     061A  1217 ;
                     061A  1218 ;               WRITE CHECK
                     061A  1219 ;               WRITE DATA
                     061A  1220 ;               READ DATA, AND
                     061A  1221 ;               READ HEADER
                     061A  1222 ;
                     061A  1223 ; INPUTS:
                     061A  1224 ;       R3      - CASE INDEX
                     061A  1225 ;       R4      - DEVICE CSR ADDRESS
                     061A  1226 ;       R5      - UCB ADDRESS
                     061A  1227 ;
                     061A  1228 ; FUNCTIONAL DESCRIPTION:
                     061A  1229 ;
                     061A  1230 ; A UNIBUS DATAPATH IS REQUESTED FOLLOWED BY THE APPROPRIATE NUMBER OF MAP
                     061A  1231 ; REGISTERS REQUIRED FOR THE TRANSFER. THE TRANSFER PARAMETERS ARE LOADED
                     061A  1232 ; INTO THE DEVICE REGISTERS, INTERRUPTS ARE LOCKED OUT, THE FUNCTION IS
                     061A  1233 ; INITIATED, AND A WAITFOR INTERRUPT AND KEEP CHANNEL IS EXECUTED.
                     061A  1234 ;
                     061A  1235 ; UPON RETURN FROM THE INTERRUPT SERVICE ROUTINE, IF THE TRANSFER IS
                     061A  1236 ; COMPLETE, THE APPROPRIATE EXIT IS TAKEN. IF THE FUNCTION IS NOT COMPLETE
                     061A  1237 ; TRANSFER PARAMETERS ARE UPDATED AND A RETURN TO FDISPATCH IS EXECUTED TO
                     061A  1238 ; RE-ISSUE SEEK AND TRANSFER FUNCTIONS WHILE KEEPING CHANNEL AND UBA
                     061A  1239 ; RESOURCES. IF A DATA CHECK HAS BEEN REQUESTED, IT IS PERFORMED
                     061A  1240 ; BEFORE RETURNING TO FDISPATCH.
                     061A  1241 ;
                     061A  1242
                     061A  1243 XFER:                                           ;TRANSFER FUNCTION EXECUTION
          01   E0    061A  1244          BBS     #UCB$V_DL_MAPPING,-           ;ADAPTER MAPPING?
        3E 00F6 C5    061C  1245                  UCB$W_DL_FLAGS(R5),2$         ;BRANCH IF ADAPTER MAPPING.
00EC C5   00F2 C5 B0  0620  1246          MOVW    UCB$A_DL_BUF_PA(R5),UCB$W_DL_SBA(R5);GET 1ST WORD OF BUFFER ADDR
    50    00F4 C5 3C  0627  1247          MOVZWL  UCB$A_DL_BUF_PA+2(R5),R0;GET BITS 16:21 OF BUFFER ADDRESS
      08  A4   50 B0  062C  1248          MOVW    R0,RL_BAE(R4)                 ;SET MEMORY EXTENSION BITS IN BAE
    50    50   04 78  0630  1249          ASHL    #4,R0,R0                      ;PUT MEMORY EXTENSION BITS IN <5:4>
    00EB  C5   50 90  0634  1250          MOVB    R0,UCB$B_DL_XBA(R5)           ;OF CSR
                     0639  1251 ;
                     0639  1252 ; FIRST TRANSFER OF THIS I/O REQUEST - ALLOCATE RESOURCES
                     0639  1253 ;
          00D6 C5    B5    0639  1254          TSTW    UCB$W_DL_DPN(R5)          ;RESOURCES ALREADY ALLOCATED?
               61    12    063D  1255          BNEQ    5$                        ;IF NEQ - YES
          00E0 C5    D4    063F  1256          CLRL    UCB$A_DL_MOVRTN(R5)       ;ASSUME READ
          53   0B    91    0643  1257          CMPB    #CDF_WRITEDATA,R3         ;WRITE DATA?
               09    12    0646  1258          BNEQ    1$                        ;IF NEQ NO
     00000000'GF     9E    0648  1259          MOVAB   G^IOC$MOVFRUSER,-         ;SET MOVE ROUTINE ADDRESS FOR
          00E0 C5          064E  1260                  UCB$A_DL_MOVRTN(R5)       ;1ST PARTIAL WRITE
00D8 C5     78 A5 D0      0651  1261 1$:      MOVL    UCB$L_SVAPTE(R5),UCB$L_DL_SVAPTE(R5);SAVE SVAPTE FOR BUFFER COPY
   00D6 C5    01 AE       0657  1262          MNEGW   #1,UCB$W_DL_DPN(R5)       ;SET FIRST XFER FLAG
               42 11       065C  1263          BRB     5$                        ;JOIN COMMON CODE
                     065E  1264
                     065E  1265 ;
                     065E  1266 ; FIRST TRANSFER OF THIS I/O REQUEST - ALLOCATE RESOURCES
                     065E  1267 ;
          00D6 C5    B5    065E  1268 2$:      TSTW    UCB$W_DL_DPN(R5)          ;UBA RESOURCES ALREADY ALLOCATED?
```

```
                     3C   12  0662  1269          BNEQ    5$                          ;IF NEQ - YES
                                0664  1270          REQDPR                              ;REQUEST DATAPATH
                                066A  1271          REQMPR                              ;REQUEST MAP REGISTERS
                                0670  1272          LOADUBA                             ;LOAD UNIBUS MAP REGISTERS
              51    24 A5   D0  0676  1273          MOVL    UCB$L_CRB(R5),R1            ;GET CRB ADDRESS
                 05    00   EF  067A  1274          EXTZV   #VEC$V_DATAPATH,#VEC$S_DATAPATH,-  ;EXTRACT DATAPATH NUMBER -
              50    37 A1         067D  1275                  CRB$L_INTD+VEC$B_DATAPATH(R1),R0   ;FOR UBA RESOURCE FLAG
           00D6 C5    50   B0  0680  1276          MOVW    R0,UCB$W_DL_DPN(R5)         ;INDICATE UBA RESOURCES ALLOCATED
                                0685  1277
              50    7C A5   3C  0685  1278          MOVZWL  UCB$W_BOFF(R5),R0           ;GET BYTE OFFSET IN PAGE
                    34 A1   F0  0689  1279          INSV    CRB$L_INTD+VEC$W_MAPREG(R1),-  ;INSERT HIGH 7 BITS OF ADDRESS
           50    07    09         068C  1280                  #9,#7,R0                    ;
           00EC C5    50   B0  068F  1281          MOVW    R0,UCB$W_DL_SBA(R5)         ;SET BUFFER ADDRESS
     50    34 A1    02   07   EF  0694  1282          EXTZV   #7,#2,CRB$L_INTD+VEC$W_MAPREG(R1),R0  ;GET MEMORY EXTENSION BITS
        00EB C5    50   10   85  069A  1283          MULB3   #16,R0,UCB$B_DL_XBA(R5) ;POSITION MEMORY EXTENSION BITS TO <5:4>
                                06A0  1284
                                06A0  1285 ;
                                06A0  1286 ; COMMON TRANSFER POINT
                                06A0  1287 ;
                                06A0  1288
                                06A0  1289 ;
                                06A0  1290 ; FOR A READ OPERATION WHEN NO ADAPTER MAPPING IS PRESENT EMPTY THE
                                06A0  1291 ; INTERNAL PHYSICALLY CONTIGUOUS BUFFER FROM THE PREVIOUS READ TO THE
                                06A0  1292 ; USER'S BUFFER.
                                06A0  1293 ;
                     02DA   30  06A0  1294 5$:      BSBW    DL_MOVE_TO_BUFFER           ;COPY TO USER BUFFER
                                06A3  1295 ;
                                06A3  1296 ; PUT BUFFER ADDRESS, WORD COUNT, AND DISK ADDRESS IN DEVICE REGISTERS
                                06A3  1297 ;
                                06A3  1298
        02 A4    00EC C5   B0  06A3  1299          MOVW    UCB$W_DL_SBA(R5),RL_BA(R4)  ;SET BUFFER ADDRESS
                 00C0 C5   AE  06A9  1300          MNEGW   UCB$W_BCR(R5),-             ;GET BYTES LEFT TO TRANSFER AND -
                 00CC C5         06AD  1301                  UCB$W_DL_PBCR(R5)           ;ASSUME ONLY ONE TRANSFER NEEDED
              52    44 A5   9A  06B0  1302          MOVZBL  UCB$B_SECTORS(R5),R2        ;GET SECTORS/SURFACE
           51    00BC C5   9A  06B4  1303          MOVZBL  UCB$W_DA(R5),R1             ;GET DESIRED SECTOR
                 52    51  A2  06B9  1304          SUBW    R1,R2                       ;CALCULATE SECTORS LEFT ON SURFACE
              52   0100 8F   A4  06BC  1305          MULW    #256,R2                     ;CONVERT TO BYTES LEFT ON SURFACE
              52   00CC C5   B1  06C1  1306          CMPW    UCB$W_DL_PBCR(R5),R2        ;ARE ADDITIONAL TRANSFERS REQUIRED?
                        05   1B  06C6  1307          BLEQU   10$                         ;IF LEQU - NO
           00CC C5    52   B0  06C8  1308          MOVW    R2,UCB$W_DL_PBCR(R5)        ;SET BYTE COUNT FOR THIS TRANSFER
                                06CD  1309 ;
                                06CD  1310 ; FOR A WRITE OPERATION WHEN NO ADAPTER MAPPING IS PRESENT
                                06CD  1311 ; FILL INTERNAL PHYSICALLY CONTIGUOUS BUFFER FROM THE USER'S BUFFER.
                                06CD  1312 ;
                     02F2   30  06CD  1313 10$:     BSBW    DL_MOVE_FROM_BUFFER         ;COPY FROM USER BUFFER
                                06D0  1314
              50    00EB C5   9A  06D0  1315          MOVZBL  UCB$B_DL_XBA(R5),R0         ;SET MEMORY EXTENSION BITS
        50    F95E CF43   A8  06D5  1316          BISW    FTAB[R3],R0                 ;MERGE XBA BITS WITH FUNCTION
     52    00CC C5    02   A7  06DB  1317          DIVW3   #2,UCB$W_DL_PBCR(R5),R2     ;CALCULATE TRANSFER WORD COUNT
              06 A4    52   AE  06E1  1318          MNEGW   R2,RL_MP(R4)                ;SET TRANSFER WORD COUNT
                                06E5  1319
              51    00BC C5   9A  06E5  1320          MOVZBL  UCB$W_DA(R5),R1             ;PUT DESIRED SECTOR IN R1<5:0>
51    01    06    00BD C5   F0  06EA  1321          INSV    UCB$W_DA+1(R5),#6,#1,R1     ;INSERT DESIRED SURFACE IN R1<6>
51    09    07    00BE C5   F0  06F1  1322          INSV    UCB$W_DC(R5),#7,#9,R1       ;INSERT DESIRED CYLINDER IN R1<15:7>
              04 A4    51   B0  06F8  1323          MOVW    R1,RL_DA(R4)                ;SET DESIRED DISK ADDRESS
                                06FC  1324
                                06FC  1325 ;
```

```
                                06FC  1326 ; EXECUTE THE TRANSFER FUNCTION
                                06FC  1327 ;
                                06FC  1328
                                06FC  1329          CKPWR                               ;DISABLE INTERRUPTS, CHECK POWER,-
                                0715  1330                                              ;AND PUT UNIT NUMBER IN R2<9:8>
                64   50   52   A9  0715  1331          BISW3   R2,R0,RL_CS(R4)             ;EXECUTE FUNCTION
                                0719  1332          WFIKPCH RETREG,#6                   ;WAITFOR INTERRUPT AND KEEP CHANNEL
                                0723  1333                                              ;RETURN HERE FROM ISR SAVING REGISTERS
                                0723  1334          IOFORK                              ;CREATE FORK PROCESS (RETURN TO ISR)
                                0729  1335                                              ;RETURN HERE FROM ISR REI ROUTINE
                                0729  1336
                                0729  1337 ;
                                0729  1338 ; PURGE DATAPATH
                                0729  1339 ;
                                0729  1340
                00E4 C5   94    0729  1341          CLRB    UCB$B_DL_DPPE(R5)           ;CLEAR DATAPATH PURGE ERROR
          00000000'GF   16      072D  1342          JSB     G^IOC$PURGDATAP             ;PURGE DATAPATH
                04 50   E8      0733  1343          BLBS    R0,20$                      ;IF SET - NO PURGE ERRORS
                00E4 C5   96    0736  1344          INCB    UCB$B_DL_DPPE(R5)           ;SET DATAPATH PURGE ERROR
                                073A  1345
                                073A  1346 ;
                                073A  1347 ; SAVE UBA REGISTERS FOR UPDATE AND REGDUMP ROUTINES
                                073A  1348 ;
                                073A  1349
                      01   E1   073A  1350 20$:     BBC     #UCB$V_DL_MAPPING,-         ;ADAPTER MAPPING?
                3E 00F6 C5         073C  1351                  UCB$W_DL_FLAGS(R5),30$  ;IF BC NO
                00D8 C5   51   D0  0740  1352          MOVL    R1,UCB$L_DL_DPR(R5)         ;SAVE DATAPATH REGISTER
     50   00D0 C5   07   09   EF  0745  1353          EXTZV   #9,#7,UCB$W_DL_BA(R5),R0    ;EXTRACT LOW BITS OF FINAL MAP REG NO.
     51   00CE C5   02   04   EF  074C  1354          EXTZV   #4,#2,UCB$W_DL_CS(R5),R1    ;EXTRACT HI BITS OF FINAL MAP REG NO.
          50   02   07   51   F0  0753  1355          INSV    R1,#7,#2,R0                 ;INSERT HIGH BITS OF FINAL MAP REGISTER
               50   01EF 8F   B1  0758  1356          CMPW    #495,R0                     ;LEGAL MAP REGISTER NUMBER?
                         05   18  075D  1357          BGEQ    25$                         ;IF GEQ - YES
               50   01EF 8F   3C  075F  1358          MOVZWL  #495,R0                     ;RESTRICT MAP REGISTER NUMBER
     00DC C5   6240   D0  0764  1359 25$:     MOVL    (R2)[R0],UCB$L_DL_FMPR(R5)  ;SAVE FINAL MAP REGISTER NUMBER
                00E0 C5   D4  076A  1360          CLRL    UCB$L_DL_PMPR(R5)           ;CLEAR PREVIOUS MAP REGISTER CONTENTS
                     50   D7  076E  1361          DECL    R0                          ;CALCULATE PREVIOUS MAP REGISTER NUMBER
                0F   00   EC  0770  1362          CMPV    #VEC$V_MAPREG,#VEC$S_MAPREG,-  ;ANY PREVIOUS MAP REGISTER?
          50   34 A3         0773  1363                  CRB$L_INTD+VEC$W_MAPREG(R3),R0 ;...
                     06   14  0776  1364          BGTR    30$                         ;IF GTR - NO
     00E0 C5   6240   D0  0778  1365          MOVL    (R2)[R0],UCB$L_DL_PMPR(R5)  ;SAVE PREVIOUS MAP REGISTER
03   00CE C5     0F   E1  077E  1366 30$:     BBC     #RL_CS_V_CE,UCB$W_DL_CS(R5),40$  ;IF CLR - NO RL ERRORS
               0098   31  0784  1367          BRW     RETREG                      ;DEVICE ERROR
          03 00E4 C5   E9  0787  1368 40$:     BLBC    UCB$B_DL_DPPE(R5),45$       ;IF CLR - NO PURGE ERROR
               0090   31  078C  1369          BRW     RETREG                      ;PURGE ERROR
                                078F  1370
                                078F  1371 ;
                                078F  1372 ; RETURN HEADER INFORMATION FOR READ HEADER FUNCTION
                                078F  1373 ;
                                078F  1374
     0093 C5     0E   91  078F  1375 45$:     CMPB    #CDF_READHEAD,UCB$B_CEX(R5)  ;READ HEADER FUNCTION?
                      2F   12  0794  1376          BNEQ    DATACHECK                   ;IF NEQ - NO
                00C0 C5   DD  0796  1377          PUSHL   UCB$W_BCR(R5)               ;SAVE NEG BYTES REMAINING
                   78 A5   DD  079A  1378          PUSHL   UCB$L_SVAPTE(R5)            ;SAVE ADDRESS OF PTE
     51   00E5 C5   9E  079D  1379          MOVAB   UCB$W_DL_DB(R5),R1          ;SET ADDRESS OF INTERNAL BUFFER
          52   06   D0  07A2  1380          MOVL    #6,R2                       ;SET NUMBER OF BYTES TO MOVE
       7E A5   52   B1  07A5  1381          CMPW    R2,UCB$W_BCNT(R5)           ;ROOM FOR FULL HEADER?
                      04   1F  07A9  1382          BLSSU   50$                         ;IF LSSU - YES
```

```
                52   7E A5    3C  07AB  1383          MOVZWL  UCB$W_BCNT(R5),R2         ;SET LENGTH OF PARTIAL HEADER
        00C0 C5 52   7E A5    A3  07AF  1384 50$:      SUBW3   UCB$W_BCNT(R5),R2,UCB$W_BCR(R5)  ;CALCULATE TRANSFER BYTE COUNT
                00000000'GF    16  07B6  1385          JSB     G^IOC$MOVTOUSER             ;MOVE HEADER TO USER BUFFER
                    78 A5 8ED0  07BC  1386          POPL    UCB$L_SVAPTE(R5)            ;RESTORE ADDRESS OF PTE
                    00C0 C5 8ED0  07C0  1387          POPL    UCB$W_BCR(R5)               ;RESTORE NEG BYTES REMAINING
                                07C5  1388
                                07C5  1389 ;
                                07C5  1390 ; PERFORM DATA CHECK, IF REQUESTED
                                07C5  1391 ;
                                07C5  1392
                                07C5  1393 DATACHECK:                                   ;DATACHECK AFTER PARTIAL TRANSFER
                        0E   E1 07C5  1394          BBC     #IO$V_DATACHECK,-           ;IF CLR - DATA CHECK NOT REQUESTED
                  10 009A C5    07C7  1395                  UCB$W_FUNC(R5),UPDATE       ;...
                00C9 C5   00 E4 07CB  1396          BBSC    #0,UCB$B_DL_DCHEK(R5),-     ;IF SET - DATA CHECK ALREADY PERFORMED
                        0A      07D0  1397                  UPDATE                      ;...
                    00C9 C5   96  07D1  1398          INCB    UCB$B_DL_DCHEK(R5)          ;SET DATA CHECK IN PROGRESS
                    53   0A   9A  07D5  1399          MOVZBL  #IO$_WRITECHECK,R3          ;SET CASE INDEX TO WRITE CHECK
                       FE3F  31  07D8  1400          BRW     XFER                        ;BRANCH TO PERFORM WRITE CHECK
                                07DB  1401
                                07DB  1402 ;
                                07DB  1403 ; UPDATE BUFFER ADDRESS, CURRENT DISK ADDRESS, AND BYTES REMAINING
                                07DB  1404 ; FOR NEXT TRANSFER
                                07DB  1405 ;
                                07DB  1406
                                07DB  1407 UPDATE:                                      ;UPDATE TRANSFER PARAMETERS
                        01   E1 07DB  1408          BBC     #UCB$V_DL_MAPPING,-         ;ADAPTER MAPPING?
                  10 00F6 C5    07DD  1409                  UCB$W_DL_FLAGS(R5),10$      ;IF BC NO
                00CE C5   CF 8F   8B  07E1  1410          BICB3   #^XCF,UCB$W_DL_CS(R5),- ;SAVE MEMORY EXTENSION BITS
                    00EB C5       07E7  1411                  UCB$B_DL_XBA(R5)            ;...
                    00D0 C5   B0  07EA  1412          MOVW    UCB$W_DL_BA(R5),-           ;UPDATE SAVED BUFFER ADDRESS
                    00EC C5       07EE  1413                  UCB$W_DL_SBA(R5)            ;...
                                07F1  1414
                    00BC C5   94  07F1  1415 10$:      CLRB    UCB$W_DA(R5)                ;UPDATE DESIRED SECTOR TO ZERO
    51  00D2 C5  00000040 8F   C1  07F5  1416          ADDL3   #^0100,UCB$W_DL_DA(R5),R1   ;INCREMENT CYLINDER & SURFACE
            52  51   01   06   EF  07FF  1417          EXTZV   #6,#1,R1,R2                 ;EXTRACT DESIRED DISK SURFACE
                00BD C5   52   90  0804  1418          MOVB    R2,UCB$W_DA+1(R5)           ;UPDATE DESIRED DISK SURFACE
            52  51   09   07   EF  0809  1419          EXTZV   #7,#9,R1,R2                 ;EXTRACT DESIRED DISK CYLINDER
                00BE C5   52   B0  080E  1420          MOVW    R2,UCB$W_DC(R5)             ;UPDATE DESIRED DISK CYLINDER
                    00CC C5   A0  0813  1421          ADDW    UCB$W_DL_PBCR(R5),-         ;UPDATE NEG BYTES REMAINING TO XFER
                    00C0 C5       0817  1422                  UCB$W_BCR(R5)               ;...
                        03   13  081A  1423          BEQL    RETREG                      ;IF EQL - TRANSFER COMPLETE
                       FAF8  31  081C  1424          BRW     FDISPATCH                   ;MORE BYTES REMAINING - CONTINUE
                                081F  1425
                                081F  1426 ;
                                081F  1427 ; GET STATUS AND RESET ERRORS
                                081F  1428 ;
                                081F  1429
                                081F  1430 RETREG:                                      ;GET STATUS AND RESET ERRORS
                                081F  1431 ;
                                081F  1432 ; FOR A READ OPERATION WHEN NO ADAPTER MAPPING IS PRESENT
                                081F  1433 ; EMPTY INTERNAL BUFFER INTO USER'S BUFFER FOR LAST READ
                                081F  1434 ;
                       015B  30 081F  1435          BSBW    DL_MOVE_TO_BUFFER           ;MOVE LAST READ INTO USER'S BUFFER
                                0822  1436
                                0822  1437          SETIPL  UCB$B_FIPL(R5)              ;MAKE SURE AT FORK IPL (TIMEOUT)
                  04 A4   03   B0  0826  1438          MOVW    #RL_DA_M_STS!-              ;PUT GET STATUS IN DAR
                                082A  1439                  RL_DA_M_ARK,RL_DA(R4)       ;...
```

```
                     52   D4  082A  1440          CLRL    R2                              ;CLEAR R2 FOR UNIT NUMBER
          52  08   08  54 A5  F0  082C  1441          INSV    UCB$W_UNIT(R5),#8,#8,R2         ;GET UNIT NUMBER
              64   04  52 A9  A9  0832  1442          BISW3   R2,#F_GETSTATUS,RL_CS(R4)    ;EXECUTE GET STATUS
                     FA2A  30  0836  1443          BSBW    DL_WAIT                         ;WAIT FOR CONTROLLER
              00D4 C5  06 A4  B0  0839  1444          MOVW    RL_MP(R4),UCB$W_DL_MP(R5)    ;RETRIEVE ERROR REGISTER
                   04 A4  0B  B0  083F  1445          MOVW    #RL_DA_M_RST!-                  ;PUT GET STATUS & RESET IN DAR
                                  0843  1446                  RL_DA_M_STS!RL_DA_M_MRK,RL_DA(R4)  ;...
                  64   04  52 A9  0843  1447          BISW3   R2,#F_GETSTATUS,RL_CS(R4)    ;EXECUTE RESET
                     FA19  30  0847  1448          BSBW    DL_WAIT                         ;WAIT FOR CONTROLLER
                                  084A  1449
                                  084A  1450 ;
                                  084A  1451 ; DETERMINE EXIT - SPECIAL CONDITION, FATAL ERROR, RETRIABLE ERROR, OR SUCCESS
                                  084A  1452 ;
                                  084A  1453
                00D4 C5  05  00   ED  084A  1454          CMPZV   #0,#5,UCB$W_DL_MP(R5),- ;HEADS, BRUSHES, STATE OK?
                              1D        0850  1455                  #RL_MP_M_BH!RL_MP_M_HO!RL_SLM  ;...
                              0E  13  0851  1456          BEQL    1$                      ;IF EQL - YES, ONLINE
                64 A5   0040 8F   AA  0853  1457          BICW    #UCB$M_TIMOUT,UCB$W_STS(R5)  ;CLEAR DEVICE TIME OUT
                   50   01A4 8F   3C  0859  1458          MOVZWL  #SS$_MEDOFL,R0          ;SET MEDIUM OFFLINE STATUS
                     FB85  31  085E  1459          BRW     FUNCXT                  ;RETURN
                64 A5   0060 8F   B3  0861  1460 1$:     BITW    #UCB$M_POWER!-          ;POWER FAIL OR DEVICE TIMEOUT?
                                  0867  1461                  UCB$M_TIMOUT,UCB$W_STS(R5)  ;...
                              53  12  0867  1462          BNEQ    SPECOND                 ;IF NEQ - YES, SPECIAL CONDITION
                                  0869  1463
                4A 00D4 C5   09   E0  0869  1464          BBS     #RL_MP_V_VC,UCB$W_DL_MP(R5),20$  ;IF SET - VOLUME INVALID
                05 00CE C5   0F   E0  086F  1465          BBS     #RL_CS_V_CE,UCB$W_DL_CS(R5),2$  ;IF SET - RL ERROR
                     37 00E4 C5   E9  0875  1466          BLBC    UCB$B_DL_DPPE(R5),10$   ;IF CLR - NO PURGE ERROR
                   00000000'GF   16  087A  1467 2$:     JSB     G^ERL$DEVICERR          ;ALLOCATE AND FILL ERROR MESSAGE BUFFER
                33 009A C5   0F   E0  0880  1468          BBS     #IO$V_INHRETRY,UCB$W_FUNC(R5),20$  ;IF SET - RETRY INHIBITED
                2D 00CE C5   0D   E0  0886  1469          BBS     #RL_CS_V_NXM,UCB$W_DL_CS(R5),20$  ;IF SET - NONEXISTENT MEMORY
                15 00CE C5   0E   E1  088C  1470          BBC     #RL_CS_V_DE,UCB$W_DL_CS(R5),5$  ;IF CLR - NO DRIVE ERRORS
                06 00D4 C5   0D   E1  0892  1471          BBC     #RL_MP_V_WL,UCB$W_DL_MP(R5),4$  ;IF CLR - NOT WRITE LOCKED
                18 00D4 C5   0A   E0  0898  1472          BBS     #RL_MP_V_WGE,UCB$W_DL_MP(R5),20$  ;IF WL & WGE SET - WL ERROR
          00D4 C5   C500 8F   B3  089E  1473 4$:     BITW    #RL_MP_M_WDE!-          ;WRITE DATA ERROR, OR
                                  08A5  1474                  RL_MP_M_CHE!-           ;CURRENT HEAD ERROR, OR
                                  08A5  1475                  RL_MP_M_WGE!-           ;WRITE GATE ERROR, OR
                                  08A5  1476                  RL_MP_M_DSE,UCB$W_DL_MP(R5)  ;DRIVE SELECT ERROR?
                              12  12  08A5  1477          BNEQ    20$                     ;IF NEQ - YES
                                  08A7  1478
                                  08A7  1479 ;
                                  08A7  1480 ; RETRIABLE ERROR EXIT
                                  08A7  1481 ;
                                  08A7  1482
             7E   009C D5   98  08A7  1483 5$:     CVTBL   @UCB$L_DPC(R5),-(SP)    ;GET BRANCH DISPLACEMENT
             009C C5   BE   C0  08AC  1484          ADDL    (SP)+,UCB$L_DPC(R5)     ;CALCULATE RETURN ADDRESS - 1
                                  08B1  1485
                                  08B1  1486 ;
                                  08B1  1487 ; SUCCESSFUL OPERATION EXIT
                                  08B1  1488 ;
                                  08B1  1489
                  009C C5   D6  08B1  1490 10$:    INCL    UCB$L_DPC(R5)           ;ADJUST TO CORRECT RETURN ADDRESS
                  009C D5   17  08B5  1491          JMP     @UCB$L_DPC(R5)          ;RETURN TO DRIVER
                                  08B9  1492
                                  08B9  1493 ;
                                  08B9  1494 ; FATAL ERROR EXIT
                                  08B9  1495 ;
                                  08B9  1496
```

```
                FADC   31   08B9  1497 20$:    BRW     FATALERR                ;FATAL ERROR EXIT
                       08BC  1498
                       08BC  1499 ;
                       08BC  1500 ; SPECIAL CONDITION EXIT (POWER FAILURE OR DEVICE TIMEOUT)
                       08BC  1501 ;
                       08BC  1502
                       08BC  1503 SPECOND:
      27 64 A5    05   E0   08BC  1504         BBS     #UCB$V_POWER,UCB$W_STS(R5),PWRFAIL  ;IF SET - POWER FAILURE
                       08C1  1505                                                 ;IF CLR - DEVICE TIMEOUT
         00000000'GF   16   08C1  1506         JSB     G^ERL$DEVICTMO              ;LOG DEVICE TIMEOUT
    64 A5    0040 8F   AA   08C7  1507         BICW    #UCB$M_TIMOUT,UCB$W_STS(R5)  ;CLEAR TIMEOUT STATUS
       50    022C 8F   3C   08CD  1508         MOVZWL  #SS$_TIMEOUT,R0             ;SET DEVICE TIMEOUT STATUS
             0080 C5   97   08D2  1509         DECB    UCB$B_ERTCNT(R5)            ;ANY ERROR RETRIES REMAINING?
                  03   13   08D6  1510         BEQL    RESETXFR                    ;IF EQL - NO
                    FA3C   31   08D8  1511         BRW     FDISPATCH                   ;RETURN
                       08DB  1512
                       08DB  1513 RESETXFR:                                       ;RESET TRANSFER BYTE COUNT
          53   58 A5   D0   08DB  1514         MOVL    UCB$L_IRP(R5),R3            ;GET ADDRESS OF I/O PACKET
    00C0 C5    32 A3   AE   08DF  1515         MNEGW   IRP$W_BCNT(R3),UCB$W_BCR(R5)  ;RESET BYTE COUNT
                    FAFE   31   08E5  1516         BRW     FUNCXT                      ;EXIT
                       08E8  1517
                       08E8  1518 PWRFAIL:                                        ;POWER FAILURE
          64 A5    20   AA   08E8  1519         BICW    #UCB$M_POWER,UCB$W_STS(R5)  ;CLEAR POWER FAILURE BIT
             00D6 C5   B5   08EC  1520         TSTW    UCB$W_DL_DPN(R5)            ;ARE UCB RESOURCES ALLOCATED?
                  12   13   08F0  1521         BEQL    50$                         ;IF EQL - NO
                  01   E1   08F2  1522         BBC     #UCB$V_DL_MAPPING,-         ;ADAPTER MAPPING?
          0C 00F6 C5        08F4  1523                 UCB$W_DL_FLAGS(R5),50$      ;IF BC NO
                       08F8  1524         RELDPR                                  ;RELEASE DATA PATH
                       08FE  1525         RELMPR                                  ;RELEASE MAP REGISTERS
                       0904  1526 50$:     RELCHAN                                 ;RELEASE CHANNEL IF OWNED
          53   58 A5   D0   090A  1527         MOVL    UCB$L_IRP(R5),R3            ;GET ADDRESS OF I/O PACKET
               2C A3   7D   090E  1528         MOVQ    IRP$L_SVAPTE(R3),-          ;RESTORE TRANSFER PARAMETERS
               78 A5        0911  1529                 UCB$L_SVAPTE(R5)            ;...
                    F992   31   0913  1530         BRW     PREPROCESS                  ;RETURN TO PREPROCESS UCB FIELDS
```

```
                     0916  1532          .SBTTL  INTERRUPT SERVICE ROUTINE
                     0916  1533 ;++
                     0916  1534 ; DL$INT - RL11 INTERRUPT SERVICE ROUTINE
                     0916  1535 ;
                     0916  1536 ; FUNCTIONAL DESCRIPTION:
                     0916  1537 ;
                     0916  1538 ;       THIS ROUTINE IS ENTERED VIA A JSB INSTRUCTION WHEN AN INTERRUPT
                     0916  1539 ;       OCCURS ON AN RL11 DISK CONTROLLER. IF THE INTERRUPT IS NOT EXPECTED,
                     0916  1540 ;       THE UNSOLICITED INTERRUPT ROUTINE DISMISSES THE INTERRUPT. IF
                     0916  1541 ;       THE INTERRUPT IS EXPECTED, DEVICE REGISTERS ARE SAVED AND THE
                     0916  1542 ;       DRIVER IS CALLED AT ITS INTERRUPT RETURN ADDRESS. THE DRIVER FORKS,
                     0916  1543 ;       CAUSING A RETURN TO THIS ROUTINE, WHICH RESTORES GENERAL REGISTERS
                     0916  1544 ;       AND DISMISSES THE INTERRUPT.
                     0916  1545 ;
                     0916  1546 ; INPUTS:
                     0916  1547 ;
                     0916  1548 ;       00(SP)  - POINTER TO ADDRESS OF THE IDB
                     0916  1549 ;       04(SP)  - SAVED R0
                     0916  1550 ;       08(SP)  - SAVED R1
                     0916  1551 ;       12(SP)  - SAVED R2
                     0916  1552 ;       16(SP)  - SAVED R3
                     0916  1553 ;       20(SP)  - SAVED R4
                     0916  1554 ;       24(SP)  - SAVED R5
                     0916  1555 ;       28(SP)  - PC AT THE TIME OF THE INTERRUPT
                     0916  1556 ;       32(SP)  - PSL AT THE TIME OF THE INTERRUPT
                     0916  1557 ;
                     0916  1558 ; OUTPUTS:
                     0916  1559 ;
                     0916  1560 ;       DEVICE REGISTERS ARE SAVED, IPL IS LOWERED TO FORK LEVEL, THE
                     0916  1561 ;       INTERRUPT IS DISMISSED, ALL REGISTERS EXCEPT R0-R5 ARE PRESERVED.
                     0916  1562 ;
                     0916  1563 ;--
                     0916  1564
                     0916  1565 DL_INT::                                     ;INTERRUPT SERVICE ROUTINE
          53    9E   D0  0916  1566          MOVL    @(SP)+,R3                ;REMOVE ADDRESS OF IDB FROM STACK
          54    63   7D  0919  1567          MOVQ    (R3),R4                  ;GET ADDRESS OF CSR AND UCB
                55   D5  091C  1568          TSTL    R5                       ;IS R5 A ZERO
                39   13  091E  1569          BEQL    DL_UNSOLNT               ;IF EQL NO OWNER
                01   E5  0920  1570          BBCC    #UCB$V_INT,-             ;IF CLR - INTERRUPT NOT EXPECTED
           34 64   A5       0922  1571                  UCB$W_STS(R5),DL_UNSOLNT  ;...
                            0925  1572
      0093 C5   0E   91  0925  1573          CMPB    #CDF_READHEAD,UCB$B_CEX(R5)  ;READ HEADER FUNCTION?
                12   12  092A  1574          BNEQ    10$                      ;IF NEQ - NO
  00E5 C5    06 A4   B0  092C  1575          MOVW    RL_MP(R4),UCB$W_DL_DB(R5)    ;SAVE SECTOR HEADER INFORMATION
  00E7 C5    06 A4   B0  0932  1576          MOVW    RL_MP(R4),UCB$W_DL_DB+2(R5)  ;...
  00E9 C5    06 A4   B0  0938  1577          MOVW    RL_MP(R4),UCB$W_DL_DB+4(R5)  ;...
                            093E  1578
          52    64   9E  093E  1579 10$:     MOVAB   RL_CS(R4),R2             ;GET ADDRESS OF CONTROL STATUS REGISTER
     53   00CE C5   9E  0941  1580          MOVAB   UCB$W_DL_CS(R5),R3       ;GET ADDRESS OF REGISTER SAVE AREA
                83   82  B0  0946  1581          MOVW    (R2)+,(R3)+              ;SAVE CONTROL STATUS REGISTER
                83   82  B0  0949  1582          MOVW    (R2)+,(R3)+              ;SAVE BUFFER ADDRESS REGISTER
                83   82  B0  094C  1583          MOVW    (R2)+,(R3)+              ;SAVE DISK ADDRESS REGISTER
                83   82  B0  094F  1584          MOVW    (R2)+,(R3)+              ;SAVE MULTIPURPOSE REGISTER
                            0952  1585
        53    10 A5   7D  0952  1586 20$:     MOVQ    UCB$L_FR3(R5),R3         ;RESTORE DRIVER CONTEXT
              0C B5   16  0956  1587          JSB     @UCB$L_FPC(R5)           ;CALL DRIVER AT INTERRUPT RETURN ADDRESS
                            0959  1588
```

```
                     0959  1589 DL_UNSOLNT:                                ;UNSOLICITED INTERRUPT
3F   BA              0959  1590          POPR    #^M<R0,R1,R2,R3,R4,R5>    ;RESTORE R0-R5
     02              095B  1591          RET                               ;RETURN FROM INTERRUPT
```

```
                     095C  1593          .SBTTL  REGISTER DUMP ROUTINE
                     095C  1594 ;++
                     095C  1595 ;
                     095C  1596 ; DL_REGDUMP - REGISTER DUMP ROUTINE
                     095C  1597 ;
                     095C  1598 ; FUNCTIONAL DESCRIPTION:
                     095C  1599 ;
                     095C  1600 ; THIS ROUTINE IS CALLED TO SAVE THE DEVICE REGISTERS AND UBA RESOURCE
                     095C  1601 ; REGISTERS IN A SPECIFIED BUFFER. IT IS CALLED FROM THE DEVICE ERROR
                     095C  1602 ; LOGGING ROUTINE AND FROM THE DIAGNOSTIC BUFFER FILL ROUTINE.
                     095C  1603 ;
                     095C  1604 ; INPUTS:
                     095C  1605 ;
                     095C  1606 ;       R0      - ADDRESS OF REGISTER SAVE BUFFER
                     095C  1607 ;       R4      - ADDRESS OF DEVICE CONTROL STATUS REGISTER (CSR)
                     095C  1608 ;       R5      - ADDRESS OF UNIT CONTROL BLOCK (UCB)
                     095C  1609 ;
                     095C  1610 ; OUTPUTS:
                     095C  1611 ;
                     095C  1612 ;       THE DEVICE AND UBA REGISTERS ARE SAVED IN THE SPECIFIED BUFFER.
                     095C  1613 ;       R0 CONTAINS THE ADDRESS OF THE NEXT EMPTY LONGWORD IN THE BUFFER.
                     095C  1614 ;       ALL REGISTERS EXCEPT R1 AND R2 ARE PRESERVED.
                     095C  1615 ;
                     095C  1616 ;--
                     095C  1617
                     095C  1618 DL_REGDUMP:                             ;REGISTER DUMP ROUTINE
     80   09   D0   095C  1619          MOVL    #<RL_NUM_REGS+5>,(R0)+  ;INSERT NUMBER OF REGISTERS
51   00CE C5   DE   095F  1620          MOVAL   UCB$Q_DL_CS(R5),R1      ;GET ADDRESS OF SAVED DEVICE REGISTERS
     52   04   9A   0964  1621          MOVZBL  #RL_NUM_REGS,R2         ;GET NUMBER OF DEVICE REGISTERS TO MOVE
     80   81   3C   0967  1622 10$:     MOVZWL  (R1)+,(R0)+             ;DUMP REGISTER IN BUFFER
       FA 52   F5   096A  1623          SOBGTR  R2,10$                  ;IF GTR - STILL MORE TO MOVF
     80   81   3C   096D  1624          MOVZWL  (R1)+,(R0)+             ;DUMP DATAPATH NUMBER
     80   81   D0   0970  1625          MOVL    (R1)+,(R0)+             ;DUMP DATAPATH REGISTER
     80   81   D0   0973  1626          MOVL    (R1)+,(R0)+             ;DUMP FINAL MAP REGISTER
     80   81   D0   0976  1627          MOVL    (R1)+,(R0)+             ;DUMP PREVIOUS MAP REGISTER
     80   81   9A   0979  1628          MOVZBL  (R1)+,(R0)+             ;DUMP DATAPATH PURGE ERROR REGISTER
                05   097C  1629          RSB                             ;RETURN
                     097D  1630
```

```
                     097D  1632          .SBTTL  MOVE TO USER BUFFER ROUTINE
                     097D  1633 ;++
                     097D  1634 ;
                     097D  1635 ; DL_MOVE_TO_BUFFER - MOVE TO USER BUFFER
                     097D  1636 ;
                     097D  1637 ; FUNCTIONAL DESCRIPTION:
                     097D  1638 ;
                     097D  1639 ; THIS ROUTINE MOVES DATA BETWEEN THE PHYSICALLY CONTIGUOUS BUFFER AND
                     097D  1640 ; THE USER'S BUFFER.
                     097D  1641 ;
                     097D  1642 ; INPUTS:
                     097D  1643 ;
                     097D  1644 ;       R5 - UCB ADDRESS
                     097D  1645 ;
                     097D  1646 ; OUTPUTS:
                     097D  1647 ;
                     097D  1648 ;       DATA MOVE BETWEEN THE PHYSICALLY CONTIGUOUS BUFFER AND THE USER'S BUFFER.
                     097D  1649 ;       REGISTER'S R0,R1, AND R2 ARE DESTROYED
                     097D  1650 ;
                     097D  1651 ;--
                     097D  1652
                     097D  1653 DL_MOVE_TO_BUFFER:                          ;BUFFER MOVE ROUTINE
          01   E0    097D  1654          BBS     #UCB$V_DL_MAPPING,-       ;ADAPTER MAPPING?
  34 00F6 C5          097F  1655                  UCB$W_DL_FLAGS(R5),10$    ;IF BS YES NOTHING TO MOVE
0093 C5   0C   91     0983  1656          CMPB    #CDF_READDATA,UCB$B_CEX(R5);READ DATA OPERATION?
          2D   12     0988  1657          BNEQ    10$                       ;IF NEQ NOT A READ
00C9 C5   00   E0     098A  1658          BBS     #0,UCB$B_DL_DCHEK(R5),- ;DATA CHECK IN PROGRESS?
          27          098F  1659                  10$                       ;IF BS YES NOTHING TO MOVE
     00E0 C5   D5     0990  1660          TSTL    UCB$A_DL_MOVRTN(R5)       ;ANYTHING TO MOVE?
          22   13     0994  1661          BEQL    20$                       ;IF EQL NO
50   00DC C5   D0     0996  1662          MOVL    UCB$L_DL_BUFADR(R5),R0    ;GET USER BUFFER POINTER
51   00EE C5   D0     099B  1663          MOVL    UCB$A_DL_BUF_VA(R5),R1    ;GET PHYSICALLY CONTIGUOUS BUFFER ADDRESS
52   00CC C5   3C     09A0  1664          MOVZWL  UCB$W_DL_PBCR(R5),R2      ;GET NUMBER OF BYTES TO TRANSFER
     00E0 D5   16     09A5  1665          JSB     @UCB$A_DL_MOVRTN(R5)      ;CALL MOVE ROUTINE
00DC C5   50   D0     09A9  1666          MOVL    R0,UCB$L_DL_BUFADR(R5)    ;SAVE INTERNAL BUFFER POINTER
 00000000'GF  9E     09AE  1667          MOVAB   G^IOC$MOVTOUSER2,-        ;SET NEXT MOVE ROUTINE TO BE USED
     00E0 C5          09B4  1668                  UCB$A_DL_MOVRTN(R5)       ;
                 05   09B7  1669 10$:     RSB                               ;RETURN
                      09B8  1670
 00000000'GF  9E     09B8  1671 20$:     MOVAB   G^IOC$MOVTOUSER,-         ;SET NEXT MOVE ROUTINE TO BE USED
     00E0 C5          09BE  1672                  UCB$A_DL_MOVRTN(R5)       ;
                 05   09C1  1673          RSB                               ;RETURN
                      09C2  1674
```

```
                        09C2  1676          .SBTTL  MOVE FROM USER BUFFER ROUTINE
                        09C2  1677 ;++
                        09C2  1678 ;
                        09C2  1679 ; DL_MOVE_FROM_BUFFER - MOVE FROM USER BUFFER
                        09C2  1680 ;
                        09C2  1681 ; FUNCTIONAL DESCRIPTION:
                        09C2  1682 ;
                        09C2  1683 ; THIS ROUTINE MOVES DATA BETWEEN THE PHYSICALLY CONTIGUOUS BUFFER AND
                        09C2  1684 ; THE USER'S BUFFER.
                        09C2  1685 ;
                        09C2  1686 ; INPUTS:
                        09C2  1687 ;
                        09C2  1688 ;       R5 - UCB ADDRESS
                        09C2  1689 ;
                        09C2  1690 ; OUTPUTS:
                        09C2  1691 ;
                        09C2  1692 ;       DATA MOVE BETWEEN THE PHYSICALLY CONTIGUOUS BUFFER AND THE USER'S BUFFER.
                        09C2  1693 ;       REGISTER'S R0,R1, AND R2 ARE DESTROYED
                        09C2  1694 ;
                        09C2  1695 ;--
                        09C2  1696
                        09C2  1697 DL_MOVE_FROM_BUFFER:                     ;BUFFER MOVE ROUTINE
          01    E0      09C2  1698          BBS     #UCB$V_DL_MAPPING,-     ;ADAPTER MAPPING?
       2E 00F6  C5      09C4  1699                  UCB$W_DL_FLAGS(R5),10$  ;IF BS YES NOTHING TO MOVE
0093   C5    08    91    09C8  1700          CMPB    #CDF_WRITEDATA,UCB$B_CEX(R5);WRITE DATA OPERATION?
          27    12      09CD  1701          BNEQ    10$                     ;IF NEQ NOT A WRITE
00C9   C5    00    E0    09CF  1702          BBS     #0,UCB$B_DL_DCHEK(R5),- ;DATA CHECK IN PROGRESS?
          21            09D4  1703                  10$                     ;IF BS YES NOTHING TO MOVE
50     00DC  C5    D0    09D5  1704          MOVL    UCB$L_DL_BUFADR(R5),R0  ;GET USER BUFFER POINTER
51     00EE  C5    D0    09DA  1705          MOVL    UCB$A_DL_BUF_VA(R5),R1  ;GET PHYSICALLY CONTIGUOUS BUFFER ADDRESS
52     00CC  C5    3C    09DF  1706          MOVZWL  UCB$W_DL_PBCR(R5),R2    ;GET NUMBER OF BYTES TO TRANSFER
          00E0  D5    16    09E4  1707          JSB     @UCB$A_DL_MOVRTN(R5)    ;CALL MOVE ROUTINE
00DC   C5    50    D0    09E8  1708          MOVL    R0,UCB$L_DL_BUFADR(R5)  ;SAVE INTERNAL BUFFER POINTER
 00000000'GF    9E    09ED  1709          MOVAB   G^IOC$MOVFRUSER2,-      ;SET NEXT MOVE ROUTINE TO BE USED
          00E0  C5          09F3  1710                  UCB$A_DL_MOVRTN(R5)     ;
                      05    09F6  1711 10$:     RSB                         ;RETURN
                            09F7  1712
                            09F7  1713 DL_END:                              ;ADDRESS OF LAST LOCATION IN DRIVER
                            09F7  1714          .END
```

```
$$$                     = 00000020 R      02
$$BASE                  = 00000001
$$DISPL                 = 0000000A
$$GENSW                 = 00000001
$$HIGH                  = 00000009
$$LIMIT                 = 00000008
$$LOW                   = 00000001
$$MNSW                  = 00000001
$$MXSW                  = 00000001
$$OP                    = 00000002
ACP$ACCESS                ********    X   03
ACP$DEACCESS              ********    X   03
ACP$MODIFY                ********    X   03
ACP$MOUNT                 ********    X   03
ACP$READBLK               ********    X   03
ACP$WRITEBLK              ********    X   03
ATS_UBA                 = 00000001
AVAILABLE                 0000036A R      03
BUG$_UNSUPRTCPU           ********    X   03
CDF_AVAILABLE           = 00000011
CDF_DRVCLR              = 00000004
CDF_NOP                 = 00000010
CDF_OFFSET              = 00000006
CDF_PACKACK             = 00000008
CDF_READDATA            = 0000000C
CDF_READHEAD            = 0000000E
CDF_RECAL               = 00000003
CDF_RELEASE             = 00000005
CDF_RETCENTER           = 00000007
CDF_SEARCH              = 00000009
CDF_SEEK                = 00000002
CDF_UNLOAD              = 00000001
CDF_WRITECHECK          = 0000000A
CDF_WRITEDATA           = 0000000B
CDF_WRITEHEAD           = 0000000D
CRB$B_MASK              = 0000000E
CRB$L_AUXSTRUC          = 00000010
CRB$L_INTD              = 00000024
CRB$V_BSY               = 00000000
DATACHECK                 000007C5 R      03
DC$_DISK                = 00000001
DDB$K_CART              = 00000002
DDB$L_ACPD              = 00000010
DDB$L_DDT               = 0000000C
DEV$M_AVL               = 00040000
DEV$M_DIR               = 00000008
DEV$M_ELG               = 00400000
DEV$M_FOD               = 00004000
DEV$M_IDV               = 04000000
DEV$M_NNM               = 00000200
DEV$M_ODV               = 08000000
DEV$M_RND               = 10000000
DEV$M_SHR               = 00010000
DL$DDT                    00000000 RG     03
DL_ALIGN                  00000298 R      03
DL_END                    000009F7 R      03
DL_FUNCTABLE              0000005C R      03
DL_INT                    00000916 RG     03
DL_MOVE_FROM_BUFFER       000009C2 R      03
DL_MOVE_TO_BUFFER         0000097D R      03
DL_REGDUMP                0000095C R      03
DL_RL0X_INIT              00000143 R      03
DL_RL11_INIT              000000FC R      03
DL_STARTIO                000002A8 R      03
DL_UNSOLNT                00000959 R      03
DL_WAIT                   00000263 R      03
DO_FUNCTION               0000035C R      03
DPT$C_LENGTH            = 00000038
DPT$C_VERSION           = 00000004
DPT$INITAB                00000038 R      02
DPT$M_SVP               = 00000002
DPT$REINITAB              00000073 R      02
DPT$TAB                   00000000 R      02
DRCLR                     00000477 R      03
DRVCLR                    0000035C R      03
DT$_RL01                = 00000009
DT$_RL02                = 0000000A
DYN$C_CRB               = 00000005
DYN$C_DDB               = 00000006
DYN$C_DPT               = 0000001E
DYN$C_UCB               = 00000010
EMB$L_DV_REGSAV         = 0000004E
ERL$DEVICERR              ********    X   03
ERL$DEVICTMO              ********    X   03
EXE$ABORTIO               ********    X   03
EXE$ALOPHYCNTG            ********    X   03
EXE$GB_CPUDATA            ********    X   03
EXE$GB_CPUTYPE            ********    X   03
EXE$GL_TENUSEC            ********    X   03
EXE$GL_UBDELAY            ********    X   03
EXE$IOFORK                ********    X   03
EXE$LCLDSKVALID           ********    X   03
EXE$ONEPARM               ********    X   03
EXE$PWRTIMCHK             ********    X   03
EXE$SENSEMODE             ********    X   03
EXE$SETCHAR               ********    X   03
EXE$ZEROPARM              ********    X   03
FATALERR                  00000398 R      03
FDISPATCH                 00000317 R      03
FEXL                      00000439 R      03
FTAB                      00000038 R      03
FUNCTAB_LEN             = 000000A0
FUNCXT                    000003E6 R      03
F_AVAILABLE             = 00000000
F_DRVCLR                = 00000004
F_GETSTATUS             = 00000004
F_NOP                   = 00000000
F_OFFSET                = 00000000
F_PACKACK               = 00000004
F_READDATA              = 0000000C
F_READHEAD              = 00000008
F_RECAL                 = 00000000
F_RELEASE               = 00000000
F_RETCENTER             = 00000000
```

```
F_SEARCH                =  00000000
F_SEEK                  =  00000006
F_UNLOAD                =  00000000
F_WRITECHECK            =  00000002
F_WRITEDATA             =  0000000A
F_WRITEHEAD             =  00000000
IDB$L_CSR               =  00000000
IDB$L_OWNER             =  00000004
IMMED                      0000047B R      03
IO$M_DATACHECK          =  00004000
IO$V_DATACHECK          =  0000000E
IO$V_INHRETRY           =  0000000F
IO$_ACCESS              =  00000032
IO$_ACPCONTROL          =  00000038
IO$_AVAILABLE           =  00000011
IO$_CREATE              =  00000033
IO$_DEACCESS            =  00000034
IO$_DELETE              =  00000035
IO$_DRVCLR              =  00000004
IO$_MODIFY              =  00000036
IO$_MOUNT               =  00000039
IO$_NOP                 =  00000000
IO$_PACKACK             =  00000008
IO$_READHEAD            =  0000000E
IO$_READLBLK            =  00000021
IO$_READPBLK            =  0000000C
IO$_READVBLK            =  00000031
IO$_SEEK                =  00000002
IO$_SENSECHAR           =  0000001B
IO$_SENSEMODE           =  00000027
IO$_SETCHAR             =  0000001A
IO$_SETMODE             =  00000023
IO$_UNLOAD              =  00000001
IO$_VIRTUAL             =  0000003F
IO$_WRITECHECK          =  0000000A
IO$_WRITELBLK           =  00000020
IO$_WRITEPBLK           =  0000000B
IO$_WRITEVBLK           =  00000030
IOC$DIAGBUFILL             ********    X    03
IOC$LOADUBAMAP             ********    X    03
IOC$MNTVER                 ********    X    03
IOC$MOVFRUSER              ********    X    03
IOC$MOVFRUSER2             ********    X    03
IOC$MOVTOUSER              ********    X    03
IOC$MOVTOUSER2             ********    X    03
IOC$PURGDATAP              ********    X    03
IOC$RELCHAN                ********    X    03
IOC$RELDATAP               ********    X    03
IOC$RELMAPREG              ********    X    03
IOC$REQCOM                 ********    X    03
IOC$REQDATAP               ********    X    03
IOC$REQMAPREG              ********    X    03
IOC$REQPCHANL              ********    X    03
IOC$RETURN                 ********    X    03
IOC$WFIKPCH                ********    X    03
IRP$L_MEDIA             =  00000038
IRP$L_SVAPTE            =  0000002C
IRP$S_FCODE             =  00000006
IRP$V_DIAGBUF           =  00000007
IRP$V_FCODE             =  00000000
IRP$V_PHYSIO            =  00000008
IRP$W_BCNT              =  00000032
IRP$W_FUNC              =  00000020
IRP$W_STS               =  0000002A
MASKH                   =  00000008
MASKL                   =  04000000
MMG$GL_SPTBASE             ********    X    03
NOP                        0000035C R      03
NORMAL                     00000389 R      03
PACKACK                    00000362 R      03
POSIT                      000004AE R      03
PR$_IPL                 =  00000012
PR$_SID_TYP730          =  00000003
PR$_SID_TYP750          =  00000002
PR$_SID_TYP780          =  00000001
PR$_SID_TYP785          =  00000009
PR$_SID_TYP790          =  00000004
PR$_SID_TYPMAX          =  00000008
PR$_SID_TYPUV1          =  00000007
PREPROCESS                 000002A8 R      03
PTE$S_PFN               =  00000015
PWRFAIL                    000008E8 R      03
READDATA                   00000379 R      03
READHEAD                   00000372 R      03
RESETXFR                   000008DB R      03
RETREG                     0000081F R      03
RETRYERP                   0000038F R      03
RL_BA                      00000002
RL_BAE                     00000008
RL_CS                      00000000
RL_CS_M_CE              =  00008000
RL_CS_M_CRDY            =  00000080
RL_CS_M_DRDY            =  00000001
RL_CS_M_IE              =  00000040
RL_CS_V_CE              =  0000000F
RL_CS_V_CRC             =  0000000B
RL_CS_V_DE              =  0000000E
RL_CS_V_NXM             =  0000000D
RL_CS_V_OPI             =  0000000A
RL_DA                      00000004
RL_DA_M_MRK             =  00000001
RL_DA_M_RST             =  00000008
RL_DA_M_STS             =  00000002
RL_MP                      00000006
RL_MP_M_BH              =  00000008
RL_MP_M_CHE             =  00004000
RL_MP_M_DSE             =  00000100
RL_MP_M_HO              =  00000010
RL_MP_M_TYP             =  00000080
RL_MP_M_WDE             =  00008000
RL_MP_M_WGE             =  00000400
RL_MP_V_VC              =  00000009
RL_MP_V_WGE             =  0000000A
RL_MP_V_WL              =  0000000D
```

```
RL_NUM_REGS                     =  00000004
RL_SLM                          =  00000005
SEEK                               0000035C  R      03
SIZ...                          =  00000001
SPECOND                            000008BC  R      03
SS$_CTRLERR                     =  00000054
SS$_DATACHECK                   =  0000005C
SS$_DRVERR                      =  0000008C
SS$_IVBUFLEN                    =  0000034C
SS$_MEDOFL                      =  000001A4
SS$_NORMAL                      =  00000001
SS$_PARITY                      =  000001F4
SS$_TIMEOUT                     =  0000022C
SS$_VOLINV                      =  00000254
SS$_WRITLCK                     =  0000025C
UCB$A_DL_BUF_PA                    000000F2
UCB$A_DL_BUF_VA                    000000EE
UCB$A_DL_MOVRTN                    000000E0
UCB$B_CEX                       =  00000093
UCB$B_DEVCLASS                  =  00000040
UCB$B_DEVTYPE                   =  00000041
UCB$B_DIPL                      =  0000005E
UCB$B_DL_DCHEK                  =  000000C9
UCB$B_DL_DPPE                      000000E4
UCB$B_DL_XBA                       000000EB
UCB$B_ERTCNT                    =  00000080
UCB$B_ERTMAX                    =  00000081
UCB$B_FEX                       =  00000092
UCB$B_FIPL                      =  0000000B
UCB$B_SECTORS                   =  00000044
UCB$B_TRACKS                    =  00000045
UCB$K_DL_BUFSZ                  =  00000014
UCB$K_DL_LEN                       000000F8
UCB$K_LCL_DISK_LENGTH           =  000000CC
UCB$L_CRB                       =  00000024
UCB$L_DEVCHAR                   =  00000038
UCB$L_DEVCHAR2                  =  0000003C
UCB$L_DL_BUFADR                    000000DC
UCB$L_DL_DPR                       000000D8
UCB$L_DL_FMPR                      000000DC
UCB$L_DL_PMPR                      000000E0
UCB$L_DL_SVAPTE                    000000D8
UCB$L_DPC                       =  0000009C
UCB$L_FPC                       =  0000000C
UCB$L_FR3                       =  00000010
UCB$L_IRP                       =  00000058
UCB$L_MAXBLOCK                  =  000000B0
UCB$L_MEDIA                     =  000000BC
UCB$L_MEDIA_ID                  =  0000008C
UCB$L_SVAPTE                    =  00000078
UCB$M_DIAGBUF                   =  00000002
UCB$M_NOCNVRT                   =  00000004
UCB$M_ONLINE                    =  00000010
UCB$M_POWER                     =  00000020
UCB$M_TIMOUT                    =  00000040
UCB$M_VALID                     =  00000800
UCB$V_DL_22BIT                  =  00000000
UCB$V_DL_MAPPING                =  00000001
UCB$V_INT                       =  00000001
UCB$V_POWER                     =  00000005
UCB$V_VALID                     =  0000000B
UCB$W_BCNT                      =  0000007E
UCB$W_BCR                       =  000000C0
UCB$W_BOFF                      =  0000007C
UCB$W_CYLINDERS                 =  00000046
UCB$W_DA                        =  000000BC
UCB$W_DC                        =  000000BE
UCB$W_DEVBUFSIZ                 =  00000042
UCB$W_DEVSTS                    =  00000068
UCB$W_DL_BA                        000000D0
UCB$W_DL_CS                        000000CE
UCB$W_DL_DA                        000000D2
UCB$W_DL_DB                        000000E5
UCB$W_DL_DPN                       000000D6
UCB$W_DL_FLAGS                     000000F6
UCB$W_DL_MP                        000000D4
UCB$W_DL_PBCR                      000000CC
UCB$W_DL_SBA                       000000EC
UCB$W_FUNC                      =  0000009A
UCB$W_OFFSET                    =  000000C8
UCB$W_STS                       =  00000064
UCB$W_UNIT                      =  00000054
UNLOAD                             0000036A  R      03
UPDATE                             000007DB  R      03
VA$M_BYTE                       =  000001FF
VA$S_VPN                        =  00000015
VA$V_VPN                        =  00000009
VEC$B_DATAPATH                  =  00000013
VEC$L_IDB                       =  00000008
VEC$L_INITIAL                   =  0000000C
VEC$L_UNITINIT                  =  00000018
VEC$S_DATAPATH                  =  00000005
VEC$S_MAPREG                    =  0000000F
VEC$V_DATAPATH                  =  00000000
VEC$V_MAPREG                    =  00000000
VEC$W_MAPREG                    =  00000010
WRITECHECK                         00000372  R      03
WRITEDATA                          00000379  R      03
XFER                               0000061A  R      03
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	000000FA (250.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
$$$105_PROLOGUE	00000088 (136.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE
$$$115_DRIVER	000009F7 (2551.)	03 (3.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	30	00:00:00.05	00:00:00.55
Command processing	128	00:00:00.39	00:00:01.95
Pass 1	642	00:00:20.98	00:01:13.05
Symbol table sort	0	00:00:02.56	00:00:09.30
Pass 2	308	00:00:04.50	00:00:21.00
Symbol table output	38	00:00:00.23	00:00:01.81
Psect synopsis output	2	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	1150	00:00:28.74	00:01:47.68

The working set limit was 2250 pages.
171426 bytes (335 pages) of virtual memory were used to buffer the intermediate code.
There were 130 pages of symbol table space allocated to hold 2340 non-local and 91 local symbols.
1714 source lines were read in Pass 1, producing 24 object records in Pass 2.
67 pages of virtual memory were used to define 62 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SYS.OBJ]LIB.MLB;1	40
_255DUA28:[SYSLIB]STARLET.MLB;2	11
TOTALS (all libraries)	51

2659 GETS were required to define 51 macros.

There were no errors, warnings or information messages.

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

DQDRIVER
LIS

DLDRIVER
LIS

DRDRIVER
LIS

DMDRIVER
LIS