```
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      QQQQQQ    DDDDDDDD    RRRRRRRR      IIIIII    VV      VV  EEEEEEEEEE  RRRRRRRR
DDDDDDDD      QQQQQQ    DDDDDDDD    RRRRRRRR      IIIIII    VV      VV  EEEEEEEEEE  RRRRRRRR
DD      DD  QQ      QQ  DD      DD  RR      RR      II      VV      VV  EE          RR      RR
DD      DD  QQ      QQ  DD      DD  RR      RR      II      VV      VV  EE          RR      RR
DD      DD  QQ      QQ  DD      DD  RR      RR      II      VV      VV  EE          RR      RR
DD      DD  QQ      QQ  DD      DD  RR      RR      II      VV      VV  EE          RR      RR
DD      DD  QQ      QQ  DD      DD  RRRRRRRR        II      VV      VV  EEEEEEEE    RRRRRRRR
DD      DD  QQ      QQ  DD      DD  RRRRRRRR        II      VV      VV  EEEEEEEE    RRRRRRRR
DD      DD  QQ  QQ  QQ  DD      DD  RR  RR          II      VV      VV  EE          RR  RR
DD      DD  QQ  QQ  QQ  DD      DD  RR  RR          II      VV      VV  EE          RR  RR
DD      DD  QQ    QQ    DD      DD  RR    RR        II        VV  VV    EE          RR    RR        ....
DD      DD  QQ    QQ    DD      DD  RR    RR        II        VV  VV    EE          RR    RR        ....
DDDDDDDD      QQQQ  QQ  DDDDDDDD    RR      RR    IIIIII        VV      EEEEEEEEEE  RR      RR      ....
DDDDDDDD      QQQQ  QQ  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
```

```
0000     1          .TITLE  DQDRIVER - VAX/VMS RB730:RB02/RB80 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 RB730:RB02/RB80 DISK DRIVER
0000    31 ;
0000    32 ; AUTHOR:
0000    33 ;
0000    34 ; G. ROBERT     21-JAN-1981
0000    35 ;
0000    36 ; MODIFIED BY:
0000    37 ;
0000    38 ;       V03-008 RAS0300         Ron Schaefer            19-Jun-1984
0000    39 ;               Add DEV$M_NNM characteristic to DECHAR2 so that these
0000    40 ;               devices will have the "node$" prefix.
0000    41 ;
0000    42 ;       V03-007 ROW0211         Ralph O. Weber          28-DEC-1983
0000    43 ;               Change device-dependent UCB definition base from UCB$W_BCR+2
0000    44 ;               to UCB$K_LCL_DISK_LENGTH.
0000    45 ;
0000    46 ;       V03-006 PRD0035         Paul R. DeStefano       09-Sep-1983
0000    47 ;               Added EXE$LCLDSKVALID to function decision table.
0000    48 ;
0000    49 ;       V03-005 PRD0026         Paul R. DeStefano       28-Jul-1983
0000    50 ;               Modified ECC correction logic so that ECC is only applied
0000    51 ;               when there is a single bit ECC correctable error, or if there
0000    52 ;               is a multiple bit ECC correctable error and the error cannot
0000    53 ;               be corrected using retries.
0000    54 ;
0000    55 ;       V03-004 PRD0025         Paul R. DeStefano       22-Jun-1983
0000    56 ;               Modified FATALERR routine to return SS$_PARITY only for
0000    57 ;               errors that possibly indicate bad media.  All other error
```

```
0000     58 ;                  conditions which formerly returned SS$_PARITY now return
0000     59 ;                  SS$_CNTLERR.
0000     60 ;
0000     61 ;        V03-003 GRR3003         GREG ROBERT             16-SEP-1982
0000     62 ;                RECORD PREVIOUS DISK ADDRESS IN ERROR LOG BUFFERS.
0000     63 ;
0000     64 ;        V03-002 KDM0002         Kathleen D. Morse       28-Jun-1982
0000     65 ;                Added $DYNDEF.
0000     66 ;
0000     67 ;        V03-001 KTA0100         Kerbey T. Altmann       07-Jun-1982
0000     68 ;                Add code to set UCB$L_MEDIA_ID.
0000     69 ;
0000     70 ;
0000     71 ;**
```

```
0000     73          .SBTTL  PROGRAM ABSTRACT
0000     74
0000     75 ; ABSTRACT:
0000     76 ;
0000     77 ;       THIS MODULE CONTAINS THE TABLES AND ROUTINES NECESSARY TO
0000     78 ;       PERFORM ALL DEVICE-DEPENDENT PROCESSING OF AN I/O REQUEST
0000     79 ;       FOR RB730:RB02/RB80 DISK TYPES ON A VAX/VMS SYSTEM.
0000     80 ;
0000     81 ;       THE DISKS HAVE THE FOLLOWING PHYSICAL GEOMETRY:
0000     82 ;
0000     83 ;                          TRACKS/     SECTORS/      BYTES/      MAXIMUM
0000     84 ;             # CYL        CYLINDER    TRACK         SECTOR      BLOCKS
0000     85 ;
0000     86 ; RB02         512           2           40            256        20480
0000     87 ; RB80         561          14           32            512       251328
0000     88 ;
0000     89 ;       SINCE THE RB02 SECTOR SIZE IS ONLY 1/2 BLOCK, LOGICAL TO PHYSICAL
0000     90 ;       CONVERSION OF RB02 DISK ADDRESSES BY IOC$CVTLOGPHY IS DELAYED
0000     91 ;       UNTIL STARTIO IS CALLED, AND THE DISK ADDRESS IS DOUBLED PRIOR
0000     92 ;       TO CONVERSION.
0000     93 ;
0000     94 ;       ON THE RB80, THE LAST SECTOR IN EVERY TRACK IS RESERVED FOR
0000     95 ;       "SKIP SECTORING", AND THE LAST TWO CYLINDERS ARE RESERVED
0000     96 ;       FOR FIELD SERVICE.  THE USER AVAILABLE RB80 GEOMETRY IS THEREFORE:
0000     97 ;
0000     98 ; RB80         559          14           31            512       242606
0000     99 ;
0000    100 ;
0000    101 ;       THE CONTROLLER DOES NOT READ OR WRITE BEYOND THE END OF TRACK
0000    102 ;       (SPIRALLING), SO READ AND WRITE FUNCTIONS ARE BROKEN UP BY THIS
0000    103 ;       DRIVER INTO PARTIAL TRANSFERS TO THE END OF TRACK, FOLLOWED BY
0000    104 ;       AN EXPLICIT SEEK TO THE NEXT TRACK, THEN ANOTHER READ OR
0000    105 ;       WRITE FUNCTION UNTIL THE TOTAL DATA TRANSFER IS COMPLETE.
0000    106 ;       (TRACK TO TRACK SPIRALLING FOR R80'S, WITHIN A CYLINDER,
0000    107 ;       IS DONE INSIDE THE XFER ROUTINE BY WRITING THE DAR).
0000    108 ;
0000    109 ;       THE IO$_INHSEEK MODIFIER IS IGNORED BY THIS DRIVER.
0000    110 ;
0000    111 ;       THE R02 DRIVE ON AN RB730 CONTROLLER IS CALLED AN RB02.  THE
0000    112 ;       SAME DRIVE ON AN RL11 CONTROLLER IS KNOWN AS AN RL02.  SIMILARILY
0000    113 ;       THE R80 DRIVE IS KNOWN AS THE RM80, RA80, AND RB80 WHEN PLACED
0000    114 ;       ON DIFFERENT CONTROLLERS.  DRIVE DEPENDENT CHARACTERISTICS (SPEED,
0000    115 ;       SIZE, MECHANICAL TIMINGS) REMAIN THE SAME.  CONTROLLER DEPENDENT
0000    116 ;       CHARACTERISTICS (COMMANDS, COMMAND TIMINGS, ERROR REPORTING) VARY
0000    117 ;       FROM CONTROLLER TO CONTROLLER.
0000    118 ;
0000    119 ;
0000    120 ;--
```

```
                                I 12
0000    122          .SBTTL  EXTERNAL DEFINITIONS
0000    123
0000    124 ;
0000    125 ; EXTERNAL SYMBOLS
0000    126 ;
0000    127
0000    128          $ADPDEF                         ;DEFINE ADAPTER CONTROL BLOCK
0000    129          $CRBDEF                         ;DEFINE CHANNEL REQUEST BLOCK
0000    130          $DCDEF                          ;DEFINE DEVICE CLASS
0000    131          $DDBDEF                         ;DEFINE DEVICE DATA BLOCK
0000    132          $DEVDEF                         ;DEFINE DEVICE CHARACTERISTICS
0000    133          $DPTDEF                         ;DEFINE DRIVER PROLOGUE TABLE
0000    134          $DYNDEF                         ;DEFINE DYNAMIC DATA STRUCTURE TYPES
0000    135          $EMBDEF                         ;DEFINE ERROR MESSAGE BUFFER
0000    136          $IDBDEF                         ;DEFINE INTERRUPT DATA BLOCK
0000    137          $IODEF                          ;DEFINE I/O FUNCTION CODES
0000    138          $IPLDEF                         ;DEFINE IPL CODES
0000    139          $IRPDEF                         ;DEFINE I/O REQUEST PACKET
0000    140          $PRDEF                          ;DEFINE PROCESSOR REGISTERS
0000    141          $SSDEF                          ;DEFINE SYSTEM STATUS CODES
0000    142          $UBIDEF                         ;DEFINE UNIBUS ADAPTOR OFFSETS
0000    143          $UCBDEF                         ;DEFINE UNIT CONTROL BLOCK
0000    144          $VECDEF                         ;DEFINE INTERRUPT VECTOR BLOCK
0000    145
```

```
0000   147          .SBTTL  LOCAL MACRO DEFINITIONS
0000   148
0000   149 ;
0000   150 ; LOCAL MACROS
0000   151 ;
0000   152          .MACRO  REQDPRNW
0000   153                  JSB     G^IOC$REQDATAPNW
0000   154          .ENDM   REQDPRNW
0000   155
0000   156          .MACRO  LOADUBAA
0000   157                  JSB     G^IOC$LOADUBAMAPA
0000   158          .ENDM   LOADUBAA
0000   159
0000   160 ;
0000   161 ;EXFUNCL
0000   162 ; BRANCH TO SUBROUTINE WHICH REQUESTS CHANNEL (IF NOT ALREADY OWNED),
0000   163 ; EXECUTES FCODE (OR R3) FUNCTION, AND BRANCHES TO BDST ON ERROR
0000   164 ;
0000   165
0000   166          .MACRO  EXFUNCL BDST,FCODE
0000   167                  .IF NB  FCODE           ;IS FCODE NON-BLANK?
0000   168                  MOVZBL  #CD'FCODE,R3    ;IF NB - SPECIFY FCODE FUNCTION
0000   169                  .ENDC                   ;IF B - SPECIFY FNTN IN EXISTING R3
0000   170                  BSBW    FEXL            ;EXECUTE FUNCTION
0000   171                  .BYTE   BDST-.-1        ;WHERE TO GO IF ERROR
0000   172          .ENDM
0000   173
0000   174 ;
0000   175 ;GENF
0000   176 ; GENERATE FUNCTION TABLE ENTRY AND CASE TABLE INDEX SYMBOL
0000   177 ;
0000   178          .MACRO  GENF FCODE,MODS
0000   179                  CD'FCODE=.-FTAB/4
0000   180                  _TMP$VAL = FCODE
0000   181                  _TMP$VAL = _TMP$VAL ! RB_CS_M_IE
0000   182                  .IRP MODBIT,MODS
0000   183                          _TMP$VAL = _TMP$VAL ! RB_CS_M_'MODBIT'
0000   184                  .ENDR
0000   185                  .LONG   _TMP$VAL
0000   186          .ENDM
0000   187
0000   188
0000   189 ;
0000   190 ;CKPWR
0000   191 ; DISABLE INTERRUPTS, CHECK IF POWER HAS FAILED,
0000   192 ;
0000   193          .MACRO  CKPWR,DEST=RETREG,?L1
0000   194
0000   195                  SETIPL  #IPL$_POWER     ;RAISE TO POWER
0000   196                  BBC     #UCB$V_POWER,-  ;IF CLR - NO POWER FAILURE
0000   197                  UCB$W_STS(R5),L1        ;....
0000   198                  ENBINT                  ;POWER FAILURE - RETURN TO SAVED IPL
0000   199                  BRB     DEST            ;EXIT
0000   200 L1:                                      ;RETURN FOR NO POWER FAILURE
0000   201          .ENDM
0000   202
0000   203
```

```
0000    204 ;
0000    205 ;GETUNIT
0000    206 ; GET UNIT NUMBER FROM UCB, PLACE IN SPECIFIED LOCATION OR
0000    207 ; R2 BY DEFAULT
0000    208 ;
0000    209         .MACRO  GETUNIT,DEST=R2
0000    210                 CLRL    DEST                    ;CLEAR DEST FOR UNIT NUMBER
0000    211                 INSV    UCB$W_UNIT(R5), #8,#2,DEST ;PUT UNIT NUMBER IN DEST
0000    212         .ENDM
0000    213
0000    214
0000    215 ;
0000    216 ;INITIATE
0000    217 ; INITIATE A HARDWARE FUNCTION BY CLEARING CONTROLLER READY.
0000    218 ; PRESERVE THE ATTENTION AND INTERRUPT PENDING BITS BY CLEARING
0000    219 ; THEM (SINCE THEY ARE "WRITE ONES TO CLEAR" THE FOLLOWING
0000    220 ; INSTRUCTION LEAVES THEM UNMODIFIED).
0000    221 ;
0000    222         .MACRO  INITIATE
0000    223         BICL    #RB_CS_M_CRDY-          ;CLEAR CONTROLLER READY
0000    224                 !RB_CS_M_ATN-           ;...AND PRESERVE ATTENTION BITS
0000    225                 !RB_CS_M_IR,-           ;...AND INTERRUPT REQUEST BIT
0000    226                 RB_CS(R4)               ;...IN THE CSR
0000    227         .ENDM
0000    228
0000    229 ;
0000    230 ;BDRVTYP
0000    231 ; BRANCH ON DRIVE TYPE
0000    232 ;
0000    233         .MACRO  BDRVTYP TYPE,DEST
0000    234         CMPB    #DT$_'TYPE,UCB$B_DEVTYPE(R5)    ;COMPARE DRIVE TYPE
0000    235         BEQL    DEST                            ;BRANCH IF SPECIFIED TYPE
0000    236         .ENDM
0000    237
```

```
                     0000   239          .SBTTL  LOCAL SYMBOLS AND UCB EXTENSIONS
                     0000   240
                     0000   241 ;
                     0000   242 ; LOCAL SYMBOLS
                     0000   243 ;
00000007             0000   244 RB_NUM_REGS     =7                          ;NUMBER OF DEVICE REGISTERS
                     0000   245                                             ;(DOES NOT INCLUDE COMMAND REG (REG 8)
00000005             0000   246 RB_MP_C_SLM     =5                          ;STATE=SEEK LINEAR MODE (READY TO GO)
                     0000   247
                     0000   248 ;
                     0000   249 ; UCB OFFSETS WHICH FOLLOW THE STANDARD UCB FIELDS
                     0000   250 ;
                     0000   251          $DEFINI UCB                        ;START OF UCB DEFINITIONS
000000C9             0000   252 .=UCB$W_OFFSET+1                             ;REDEFINE FOR LOCAL USE
                     00C9   253
                     00C9   254 $DEF     UCB$B_DQ_FLAGS                     ;LOCAL DRIVER FLAGS
                     00C9   255          $VIELD  UCB,0,<-                   ;START OF DQ FLAGS DEFINITIONS
                     00C9   256                  <DQ_SIP,,M>,-              ; SEEK IN PROGRESS
                     00C9   257                  <DQ_DIP,,M>,-              ; DATA CHECK IN PROGRESS
                     00C9   258                  <DQ_ECC_DEFER,,M>,-        ; ECC CORRECTION DEFERRED TILL AFTER
                     00C9   259          >                                  ; RETRY ATTEMPT
                     00C9   260                                             ;END OF DQ FLAGS BIT DEFINITIONS
                     00C9   261
000000CC             00C9   262 .=UCB$K_LCL_DISK_LENGTH
                     00CC   263
                     00CC   264 ;
                     00CC   265 ;ADJACENCY OF UCB EXTENSIONS ASSUMED BY DQ_REGDUMP AND READ HEADER CODE
                     00CC   266 ;
                     00CC   267
                     00CC   268 $DEF     UCB$L_DQ_CS      .BLKL   1         ;CONTROL STATUS REGISTER
                     00D0   269 $DEF     UCB$L_DQ_BA      .BLKL   1         ;BUS ADDRESS REGISTER
                     00D4   270 $DEF     UCB$L_DQ_BC      .BLKL   1         ;BYTE COUNT REGISTER
                     00D8   271 $DEF     UCB$L_DQ_DA      .BLKL   1         ;DISK ADDRESS REGISTER
                     00DC   272 $DEF     UCB$L_DQ_MP      .BLKL   1         ;MULTIPURPOSE REGISTER
                     00E0   273 $DEF     UCB$L_DQ_FMPR    .BLKL   1         ;FINAL MAP REGISTER
                     00E4   274 $DEF     UCB$L_DQ_PMPR    .BLKL   1         ;PREVIOUS MAP REGISTER
                     00E8   275 $DEF     UCB$L_DQ_DPR     .BLKL   1         ;DATAPATH REGISTER (NEVER LOADED)
                     00EC   276 $DEF     UCB$W_DQ_HDR1    .BLKW   1         ;SAVED HEADER WORD 1
                     00EE   277 $DEF     UCB$W_DQ_HDR2    .BLKW   1         ;SAVED HEADER WORD 2
                     00F0   278 $DEF     UCB$W_DQ_HDR3    .BLKW   1         ;SAVED HEADER WORD 3
                     00F2   279 $DEF     UCB$L_DQ_CURDA   .BLKL   1         ;CURRENT DISK ADDRESS
                     00F6   280 $DEF     UCB$L_DQ_PREVDA  .BLKL   1         ;PREVIOUS DISK ADDRESS
                     00FA   281
000000FA             00FA   282 UCB$K_DQ_LEN = .                             ;LENGTH OF EXTENDED UCB
                     00FA   283          $DEFEND UCB                        ;END OF UCB DEFINITONS
                     0000   284
                     0000   285 ;
                     0000   286 ; RB730:RB02/RB80 REGISTER OFFSETS FROM CSR ADDRESS
                     0000   287 ;
                     0000   288          $DEFINI RB                         ; START OF REGISTER DEFINITIONS
                     0000   289
                     0000   290 $DEF     RB_CS            .BLKL   1         ;CONTROL STATUS REGISTER (CSR)
                     0004   291          _VIELD  RB_CS,0,<-                 ;START OF CSR BIT DEFINITIONS
                     0004   292                  <DRDY,,M>,-                ; DRIVE READY
                     0004   293                  <FCODE,3,M>,-              ; FUNCTION CODE
                     0004   294                  <,2>,-                     ; RESERVED BITS
                     0004   295                  <IE,,M>,-                  ; INTERRUPT ENABLE
```

```
0004    296                     <CRDY,,M>,-             ; CONTROLLER READY
0004    297                     <DS,2,M>,-              ; DRIVE SELECT
0004    298                     <OPI,,M>,-              ; OPERATION INCOMPLETE
0004    299                     <DCK,,M>,-              ; DATA CRC OR HEADER CRC OR DATA ECC
0004    300                     <DLT,,M>,-              ; DATA LATE OR HEADER NOT FOUND
0004    301                     <NXM,,M>,-              ; NON-EXISTENT MEMORY
0004    302                     <DE,,M>,-               ; DRIVE ERROR
0004    303                     <CE,,M>,-               ; COMPOSITE ERROR
0004    304                     <ATN,4,M>,-             ; DRIVE ATTENTION BITS
0004    305                     <ECS,2>,-               ; ECC STATUS
0004    306                     <SSEI,,M>,-             ; SKIP SECTOR ERROR INHIBIT
0004    307                     <SSE,,M>,-              ; SKIP SECTOR ERROR
0004    308                     <IR,,M>,-               ; RB730 INTERRUPT REQUEST
0004    309                     <MTN,,M>,-              ; MAINTENANCE MODE
0004    310                     <TYP,,M>,-              ; DRIVE TYPE 1=RB80, 0=RB02
0004    311                     <ASSI,,M>,-             ; AUTOMATIC SKIP SECTOR INHIBIT
0004    312                     <TOI,,M>,-              ; TIME OUT INHIBIT (U-DIAG'S)
0004    313                     <FMT,,M>,-              ; R80 FORMAT CONTROL
0004    314                     <,2>-                   ; RESERVED BITS
0004    315             >                               ;END CSR BIT DEFINITIONS
0004    316
0004    317 $DEF        RB_BA           .BLKL   1       ;BUS ADDRESS REGISTER (BAR)
0008    318
0008    319 $DEF        RB_BC           .BLKL   1       ;BYTE COUNT REGISTER (BCR)
000C    320
000C    321 $DEF        RB_DA           .BLKL   1       ;DISK ADDRESS REGISTER (DAR)
0010    322             _VIELD  RB_DA,0,<-              ;START OF DAR BIT DEFINITIONS
0010    323                     <SEC,8>,-               ; SECTOR
0010    324                     <TRK,8>,-               ; TRACK
0010    325                     <CYL,16>-               ; CYLINDER
0010    326             >                               ;END OF DAR BIT DEFINITIONS
0010    327
0010    328 $DEF        RB_MP           .BLKL   1       ;MULTIPURPOSE REGISTER (MPR)
0014    329             _VIELD  RB_MP,0,<-              ;RB02 STATUS WORD DEFINITIONS
0014    330                     <STA,3>,-               ; DRIVE STATE
0014    331                     <BH,,M>,-               ; BRUSH HOME
0014    332                     <HO,,M>,-               ; HEADS OUT
0014    333                     <CO,,M>,-               ; COVER OPEN
0014    334                     <HS,,M>,-               ; HEAD SELECT
0014    335                     <,1>,-                  ; RESERVED
0014    336                     <DSE,,M>,-              ; DRIVE SELECT ERROR
0014    337                     <VC,,M>,-               ; VOLUME CHECK
0014    338                     <WGE,,M>,-              ; WRITE GATE ERROR
0014    339                     <SPD,,M>,-              ; SPIN ERROR
0014    340                     <SKTO,,M>,-             ; SEEK TIME OUT
0014    341                     <WL,,M>,-               ; WRITE LOCK
0014    342                     <HCE,,M>,-              ; CURRENT HEAD ERROR
0014    343                     <WDE,,M>-               ; WRITE DATA ERROR
0014    344             >                               ;
0014    345             _VIELD  RB_MP,0,<-              ;GET STATUS COMMAND DEFINITIONS
0014    346                     <MRK,,M>,-              ; MARK (ALWAYS 1)
0014    347                     <STS,,M>,-              ; GET STATUS
0014    348                     <,1>,-                  ; RESERVED
0014    349                     <RST,,M>,-              ; RESET
0014    350             >
0014    351             _VIELD  RB_MP,0,<-              ;RB80 STATUS WORD DEFINITIONS
0014    352                     <SEC,5>,-               ; CURRENT RB80 SECTOR
```

```
0014   353                  <,3>,-                          ; RESERVED
0014   354                  <FLT,,M>,-                      ; DRIVE FAULT
0014   355                  <PLGV,,M>,-                     ; PLUG VALID
0014   356                  <SKE,,M>,-                      ; SEEK ERROR
0014   357                  <ONCY,,M>,-                     ; ON CYLINDER
0014   358                  <DRDY,,M>,-                     ; DRIVE READY
0014   359                  <WTP,,M>,-                      ; WRITE PROTECT
0014   360                  <,2>,-                          ; RESERVED
0014   361          >                                       ;END MPR BIT DEFINITIONS
0014   362
0014   363 $DEF     RB_EC1          .BLKL   1               ;ECC POSITION REGISTER (EPOR)
0018   364          _VIELD  RB_EC1,0,<-                     ;START OF EC1 BIT DEFINITIONS
0018   365                  <POS,13>,-                      ; STARTING BIT POSITION OF ECC ERROR
0018   366                  <,21>-                          ; RESERVED
0018   367          >                                       ;END EC1 BIT DEFINITIONS
0018   368
0018   369 $DEF     RB_EC2          .BLKL   1               ;ECC PATTERN REGISTER (EPAR)
001C   370          _VIELD  RB_EC2,0,<-                     ;START OF EC2 BIT DEFINITIONS
001C   371                  <PAT,11>,-                      ; PATTERN OF ECC ERROR BURST
001C   372                  <,21>-                          ; RESERVED
001C   373          >                                       ;END EC2 BIT DEFINITIONS
001C   374
001C   375 $DEF     RB_CMD          .BLKL   1               ;AUXILLARY COMMAND REGISTER
0020   376          _VIELD  RB_CMD,0,<-                     ;START OF CMD BIT DEFINITIONS
0020   377                  <INIT,32>,-                     ; SUBSYSTEM CLEAR <-- -1
0020   378          >                                       ;END CMD BIT DEFINITIONS
0020   379
0020   380          $DEFEND RB                              ;END RB730:RB80/RB02 REGISTER DEFS
0000   381
0000   382 ;
0000   383 ; HARDWARE FUNCTION CODES
0000   384 ;
00000000  0000   385 F_NOP=0*2                             ;NO OPERATION
00000004  0000   386 F_UNLOAD=2*2                          ;GET STATUS/RESET
00000006  0000   387 F_SEEK=3*2                            ;SEEK CYLINDER
00000006  0000   388 F_RECAL=3*2                           ;RECALIBRATE (SEEK -1)
00000004  0000   389 F_DRVCLR=2*2                          ;DRIVE CLEAR (GET STATUS)
00000000  0000   390 F_RELEASE=0*2                         ;NO OPERATION
00000000  0000   391 F_OFFSET=0*2                          ;NO OPERATION
00000000  0000   392 F_RETCENTER=0*2                       ;NO OPERATION
00000004  0000   393 F_PACKACK=2*2                         ;PACK ACKNOWLEDGE (SET VOLUME VALID)
00000000  0000   394 F_STARTSPNDL=0*2                      ;NO OPERATION
00000002  0000   395 F_WRITECHECK=1*2                      ;WRITE CHECK
0000000A  0000   396 F_WRITEDATA=5*2                       ;WRITE DATA
00000000  0000   397 F_WRITEHEAD=0*2                       ;WRITE HEADER (WHEN FMT BIT SET)
0000000C  0000   398 F_READDATA=6*2                        ;READ DATA
00000008  0000   399 F_READHEAD=4*2                        ;READ HEADER
00000004  0000   400 F_GETSTATUS=2*2                       ;GET STATUS (DRIVER INTERNAL USE)
00000000  0000   401 F_WRITETRACKD=0*2                     ;NOP
00000000  0000   402 F_READTRACKD=0*2                      ;NOP
00000004  0000   403 F_AVAILABLE=2*2                       ;GET STATUS/RESET
          0000   404
```

```
                                B 13
0000   406          .SBTTL  STANDARD TABLES
0000   407
0000   408 ;
0000   409 ; DRIVER PROLOGUE TABLE
0000   410 ;
0000   411 ;       THE DPT DESCRIBES DRIVER PARAMETERS AND I/O DATABASE FIELDS
0000   412 ;       THAT ARE TO BE INITIALIZED DURING DRIVER LOADING AND RELOADING
0000   413 ;
0000   414
0000   415          DPTAB   -                                       ;DPT CREATION MACRO
0000   416                  END=DQ_END,-                            ;END OF DRIVER LABEL
0000   417                  ADAPTER=UBA,-                           ;ADAPTER TYPE = UNIBUS
0000   418                  FLAGS=DPT$M_SVP,-                       ;SYSTEM PAGE TABLE ENTRY REQ.
0000   419                  MAXUNITS=4,-                            ;MAXIMUM FOUR DRIVES PER RB730
0000   420                  DEFUNITS=4,-                            ;INTERROGATE FOUR DRIVES
0000   421                  DELIVER=DQ_DELIVER,-                    ;UNIT TEST ROUTINE
0000   422                  UCBSIZE=UCB$K_DQ_LEN,-                  ;LENGTH OF UCB
0000   423                  NAME=DQDRIVER                           ;DRIVER NAME
0038   424
0038   425          DPT_STORE INIT                                  ;START CONTROL BLOCK INIT VALS.
0038   426          DPT_STORE DDB,DDB$L_ACPD,L,<^A\F11\>            ;DEFAULT ACP NAME
003F   427          DPT_STORE DDB,DDB$L_ACPD+3,B,DDB$K_PACK         ;ACP CLASS
0043   428          DPT_STORE UCB,UCB$B_FIPL,B,8                    ;FORK IPL
0047   429          DPT_STORE UCB,UCB$L_DEVCHAR,L,-                 ;DEVICE CHARACTERISTICS
0047   430                  <DEV$M_FOD-                     ; FILES ORIENTED
0047   431                  !DEV$M_DIR-                     ; DIRECTORY STRUCTURED
0047   432                  !DEV$M_AVL-                     ; AVAILABLE
0047   433                  !DEV$M_ELG-                     ; ERROR LOGGING
0047   434                  !DEV$M_SHR-                     ; SHAREABLE
0047   435                  !DEV$M_IDV-                     ; INPUT DEVICE
0047   436                  !DEV$M_ODV-                     ; OUTPUT DEVICE
0047   437                  !DEV$M_RND>                     ; RANDOM ACCESS
004E   438          DPT_STORE UCB,UCB$L_DEVCHAR2,L,-        ; DEVICE CHARACTERISTICS
004E   439                  <DEV$M_NNM>                     ; PREFIX NAME WITH "node$"
0055   440          DPT_STORE UCB,UCB$B_DEVCLASS,B,DC$_DISK ;DEVICE CLASS
0059   441          DPT_STORE UCB,UCB$W_DEVBUFSIZ,W,512     ;DEFAULT BUFFER SIZE
005E   442          DPT_STORE UCB,UCB$B_DIPL,B,21           ;DEVICE IPL
0062   443          DPT_STORE UCB,UCB$B_ERTMAX,B,8          ;MAX ERROR RETRY COUNT
0066   444          DPT_STORE UCB,UCB$L_DQ_CURDA,L,-1       ;CURRENT DISK ADDRESS
006D   445
006D   446          DPT_STORE REINIT                        ;START CONTROL BLOCK RE-INIT
006D   447          DPT_STORE CRB,CRB$L_INTD+4,D,DQ_INT     ;INTERRUPT SERV. ROUT. ADDRESS
0072   448          DPT_STORE CRB,CRB$L_INTD+VEC$L_INITIAL,-;CONTROLLER INIT ADDRESS
0072   449                  D,DQ_RB730_INIT                 ;...
0077   450          DPT_STORE CRB,CRB$L_INTD+VEC$L_UNITINIT,- ;UNIT INIT ADDRESS
0077   451                  D,DQ_UNIT_INIT                  ;...
007C   452          DPT_STORE DDB,DDB$L_DDT,D,DQ$DDT        ;DDT ADDRESS
0081   453
0081   454          DPT_STORE END                   ;END OF INITIALIZATION TABLE
0000   455
0000   456 ;
0000   457 ; DRIVER DISPATCH TABLE
0000   458 ;
0000   459 ;       THE DDT LISTS ENTRY POINTS FOR DRIVER SUBROUTINES WHICH ARE
0000   460 ;       CALLED BY THE OPERATING SYSTEM.
0000   461 ;
0000   462
```

```
                        463          DDTAB   -                               ;DDT CREATION MACRO
                        464                  DEVNAM=DQ,-                     ;NAME OF DEVICE
                        465                  START=DQ_STARTIO,-              ;START I/O ROUTINE
                        466                  FUNCTB=DQ_FUNCTABLE,-           ;FUNCTION DECISION TABLE
                        467                  CANCEL=0,-                      ;CANCEL=NO-OP FOR FILES DEVICE
                        468                  REGDMP=DQ_REGDUMP,-             ;REGISTER DUMP ROUTINE
                        469                  DIAGBF=<<RB_NUM_REGS+6+5+3+1>*4>,-  ;BYTES IN DIAG BUFFER
                        470                  ERLGBF=<<<RB_NUM_REGS+6+1>*4>+EMB$L_DV_REGSAV>  ;BYTES IN
                        471                                                  ;ERROR LOG BUFFER
                        472
                        473 ; DIAGNOSTIC BUFFER SIZE = <<7 RB730 REGISTER LONGWORDS + 6 UCB FIELD LONGWORDS
                        474 ;                           + 5 IOC$DIAGBUFILL LONGWORDS + 3 BUFFER ALLOCATION
                        475 ;                           LONGWORDS + 1 LONGWORD FOR # REGISTERS IN DQ_REGDUMP>
                        476 ;                           * 4 BYTES/LONGWORD>
                        477 ;
                        478 ; ERROR LOG BUFFER SIZE  = <<<7 RB730 REGISTER LONGWORDS + 6 UCB FIELD LONGWORDS
                        479 ;                           + 1 LONGWORD FOR # REGISTERS IN DQ_REGDUMP>
                        480 ;                           * 4 BYTES/LONGWORD> + BYTES NEEDED FOR ERROR LOGGER
                        481 ;                           TO SAVE SOFTWARE REGISTERS>
                        482
                        483
                        484 ;
                        485 ; HARDWARE FUNCTION CODE TABLE
                        486 ;
                        487 ;       THIS TABLE MERGES THE FUNCTION CODE BITS WITH THE
                        488 ;       INTERRUPT ENABLE BIT AND GENERATES THE CASE TABLE
                        489 ;       INDEX SYMBOL.  THIS IS AN ORDERED TABLE
                        490
0038    491 FTAB:       GENF    F_NOP                   ;NO-OP
003C    492             GENF    F_UNLOAD                ;UNLOAD VOLUME (GET STATUS/RESET)
0040    493             GENF    F_SEEK,CRDY             ;SEEK
0044    494             GENF    F_RECAL,CRDY            ;RECALIBRATE
0048    495             GENF    F_DRVCLR                ;DRIVE CLEAR (GET STATUS/RESET)
004C    496             GENF    F_RELEASE               ;RELEASE PORT (NOP)
0050    497             GENF    F_OFFSET                ;OFFSET HEADS (NOP)
0054    498             GENF    F_RETCENTER             ;RETURN HEADS TO CENTERLINE (NOP)
0058    499             GENF    F_PACKACK               ;PACK ACKNOWLEDGE (GET STATUS/RESET)
005C    500             GENF    F_STARTSPNDL            ;START SPINDLE (NOP)
0060    501             GENF    F_WRITECHECK,CRDY       ;WRITE CHECK
0064    502             GENF    F_WRITEDATA,CRDY        ;WRITE DATA
0068    503             GENF    F_READDATA,CRDY         ;READ DATA
006C    504             GENF    F_WRITEHEAD,<CRDY,FMT>  ;WRITE HEADERS
0070    505             GENF    F_READHEAD,CRDY         ;READ HEADERS
0074    506             GENF    F_WRITETRACKD           ;WRITE TRACK DESCRIPTOR (NOP)
0078    507             GENF    F_READTRACKD            ;READ TRACK DESCRIPTOR (NOP)
007C    508             GENF    F_AVAILABLE             ;SET UNIT AVAILABLE (GET STATUS/RESET)
0080    509
```

```
                        .SBTTL  FUNCTION DECISION TABLES

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

DQ_FUNCTABLE:
        FUNCTAB ,-                              ;LIST LEGAL FUNCTIONS
                <NOP,-                          ; NO-OP
                UNLOAD,-                        ; UNLOAD
                SEEK,-                          ; SEEK
                RECAL,-                         ; RECALIBRATE DRIVE
                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
                WRITEHEAD,-                     ; WRITE DISK HEADERS
                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
                RECAL,-                         ; RECALIBRATE
                DRVCLR,-                        ; DRIVE CLEAR
                PACKACK,-                       ; PACK ACKNOWLEDGE
                AVAILABLE,-                     ; AVAILABLE
                SENSECHAR,-                     ; SENSE CHARACTERISTICS
                SETCHAR,-                       ; SET CHARACTERISTICS
                SENSEMODE,-                     ; SENSE MODE
                SETMODE,-                       ; SET MODE
                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
```

```
0088   568                      MODIFY,-                ; MODIFY FILE ATTRIBUTES
0088   569                      MOUNT-                  ; MOUNT VOLUME
0088   570                      >
0090   571          FUNCTAB     +ACP$READBLK,-          ;READ FUNCTIONS
0090   572                      <READHEAD,-             ; READ HEADER
0090   573                      READLBLK,-              ; READ LOGICAL BLOCK
0090   574                      READPBLK,-              ; READ PHYSICAL BLOCK
0090   575                      READVBLK-               ; READ VIRTUAL BLOCK
0090   576                      >
009C   577          FUNCTAB     +ACP$WRITEBLK,-         ;WRITE FUNCTIONS
009C   578                      <WRITECHECK,-           ; WRITE CHECK
009C   579                      WRITEHEAD,-             ; WRITE HEADER
009C   580                      WRITELBLK,-             ; WRITE LOGICAL BLOCK
009C   581                      WRITEPBLK,-             ; WRITE PHYSICAL BLOCK
009C   582                      WRITEVBLK-              ; WRITE VIRTUAL BLOCK
009C   583                      >
00A8   584          FUNCTAB     +ACP$ACCESS,-           ;ACCESS FUNCTIONS
00A8   585                      <ACCESS,-               ; ACCEESS FILE / FIND DIRECTORY ENTRY
00A8   586                      CREATE-                 ; CREATE FILE AND/OR DIRECTORY ENTRY
00A8   587                      >
00B4   588          FUNCTAB     +ACP$DEACCESS,-         ;DEACCESS FUNCTION
00B4   589                      <DEACCESS-              ; DEACCESS FILE
00B4   590                      >
00C0   591          FUNCTAB     +ACP$MODIFY,-           ;MODIFY FUNCTIONS
00C0   592                      <ACPCONTROL,-           ; ACP CONTROL FUNCTION
00C0   593                      DELETE,-                ; DELETE FILE AND/OR DIRECTORY ENTRY
00C0   594                      MODIFY-                 ; MODIFY FILE ATTRIBUTES
00C0   595                      >
00CC   596          FUNCTAB     +ACP$MOUNT,-            ;MOUNT FUNCTION
00CC   597                      <MOUNT-                 ; MOUNT VOLUME
00CC   598                      >
00D8   599          FUNCTAB     +EXE$LCLDSKVALID,-      ;LOCAL DISK VALID FUNCTIONS
00D8   600                      <UNLOAD,-               ;UNLOAD VOLUME
00D8   601                       AVAILABLE,-            ;UNIT AVAILABLE
00D8   602                       PACKACK-               ;PACK ACKNOWLEDGE
00D8   603                      >
00E4   604          FUNCTAB     +EXE$ZEROPARM,-         ;ZERO PARAMETER FUNCTIONS
00E4   605                      <NOP,-                  ; NO-OP
00E4   606                      UNLOAD,-                ; UNLOAD
00E4   607                      RECAL,-                 ; RECALIBRATE
00E4   608                      DRVCLR,-                ; DRIVE CLEAR
00E4   609                      PACKACK,-               ; PACK ACKNOWLEDGE
00E4   610                      AVAILABLE-              ; AVAILABLE
00E4   611                      >
00F0   612          FUNCTAB     +EXE$ONEPARM,-          ;ONE PARAMETER FUNCTION
00F0   613                      <SEEK-                  ; SEEK
00F0   614                      >
00FC   615          FUNCTAB     +EXE$SENSEMODE,-        ;SENSE FUNCTIONS
00FC   616                      <SENSECHAR,-            ; SENSE CHARACTERISTICS
00FC   617                      SENSEMODE-              ; SENSE MODE
00FC   618                      >
0108   619          FUNCTAB     +EXE$SETCHAR,-          ;SET FUNCTIONS
0108   620                      <SETCHAR,-              ; SET CHARACTERISTICS
0108   621                      SETMODE-                ; SET MODE
0108   622                      >
0114   623
```

```
0114   625          .SBTTL  START I/O ROUTINE
0114   626
0114   627 ;++
0114   628 ;
0114   629 ; DQ_STARTIO - START I/O ROUTINE
0114   630 ;
0114   631 ; FUNCTIONAL DESCRIPTION:
0114   632 ;
0114   633 ;       THIS FORK PROCESS IS ENTERED FROM THE EXECUTIVE AFTER AN I/O REQUEST
0114   634 ;       PACKET HAS BEEN DEQUEUED, AND PERFORMS THE FOLLOWING:
0114   635 ;
0114   636 ;               - ACTIVATES THE DISK AFTER SETTING UCB FIELDS, OBTAINING
0114   637 ;                 UBA AND CONTROLLER RESOURCES, AND SETTING RB730 REGISTERS
0114   638 ;
0114   639 ;               - WAITS FOR AN INTERRUPT
0114   640 ;
0114   641 ;               - REGAINS CONTROL AFTER THE ISR SERVICES THE INTERRUPT, AND
0114   642 ;                       -  RE-ACTIVATES THE DISK IF THE ORIGINAL FUNCTION
0114   643 ;                          IS NOT YET COMPLETE, OR
0114   644 ;                       -  COMPLETES THE I/O REQUEST BY RELEASING RESOURCES,
0114   645 ;                          SETTING STATUS CODES, AND RETURNING TO THE EXEC.
0114   646 ;
0114   647 ; INPUTS:
0114   648 ;
0114   649 ;       R3              - IRP ADDRESS (I/O REQUEST PACKET)
0114   650 ;       R5              - UCB ADDRESS (UNIT CONTROL BLOCK)
0114   651 ;       IRP$L_MEDIA     - PARAMETER LONGWORD (LOGICAL BLOCK NUMBER)
0114   652 ;
0114   653 ; OUTPUTS:
0114   654 ;
0114   655 ;       R0      - FIRST I/O STATUS LONGWORD: STATUS CODE & BYTES XFFRED
0114   656 ;       R1      - SECOND I/O STATUS LONGWORD: 0 FOR DISKS
0114   657 ;
0114   658 ;       THE I/O FUNCTION IS EXECUTED.
0114   659 ;
0114   660 ;       ALL REGISTERS EXCEPT R0-R4 ARE PRESERVED.
0114   661 ;
0114   662 ;--
0114   663
0114   664 DQ_STARTIO:                                          ;START I/O OPERATION
00F2 C5   D0   0114   665          MOVL    UCB$L_DQ_CURDA(R5),-    ;SAVE CURRENT DISK ADDRESS
00F6 C5        0118   666                  UCB$L_DQ_PREVDA(R5)     ;... FOR ERROR LOGGING
               011B   667
               011B   668 ;
               011B   669 ;       PREPROCESS UCB FIELDS
               011B   670 ;
               011B   671
               011B   672 PREPROCESS:                                  ;ALTERNATE ENTRY NAME
  38 A3   D0   011B   673          MOVL    IRP$L_MEDIA(R3),-       ;STORE DISK ADDRESS
00BC C5        011E   674                  UCB$L_MEDIA(R5)         ;...
               0121   675          BDRVTYP RB80,TO$                ;BRANCH IF RB80
     08   E0   0127   676          BBS     #IRP$V_PHYSIO,-         ;IF SET - PHYSICAL I/O
26 2A A3       0129   677                  IRP$W_STS(R3),10$       ;...
50 00BC C5 02  C5   012C   678          MULL3   #2,UCB$L_MEDIA(R5),R0   ;RB02 HAS 1/2 SECTOR PER BLOCK
52   44 A5    9A   0132   679          MOVZBL  UCB$B_SECTORS(R5),R2    ;GET NUMBER OF SECTORS PER TRACK
        51    D4   0136   680          CLRL    R1                      ;CLEAR HIGH PART OF DIVIDEND
00BC C5 50 50 52 7B 0138   681          EDIV    R2,R0,R0,UCB$L_MEDIA(R5)  ;CALCULATE SECTOR NUMBER AND STORE
```

```
                52   45 A5    9A  013F    682          MCVZBL  UCB$B_TRACKS(R5),R2      ;GET NUMBER OF TRACKS PER CYLINDER
          51    50   50 52    7B  0143    683          EDIV    R2,R0,R0,R1              ;CALCULATE TRACK AND CYLINDER
              00BD C5   51    90  0148    684          MOVB    R1,UCB$L_MEDIA+1(R5)     ;STORE TRACK NUMBER
              00BE C5   50    B0  014D    685          MOVW    R0,UCB$L_MEDIA+2(R5)     ;STORE CYLINDER NUMBER
                                  0152    686
                    0081 C5     90  0152    687 10$:     MOVB    UCB$B_ERTMAX(R5),-       ;INITIALIZE ERROR RETRY COUNT
                    0080 C5         0156    688                  UCB$B_ERTCNT(R5)         ;...
            00C0 C5   7E A5     B0  0159    689          MOVW    UCB$W_BCNT(R5),UCB$W_BCR(R5)  ;INITIALIZE REMAINING BYTE COUNT
            009A C5   20 A3     B0  015F    690          MOVW    IRP$W_FUNC(R3),UCB$W_FUNC(R5) ;SAVE FUNCTION CODE AND MODIFIERS
                         00     EF  0165    691          EXTZV   #IRP$V_FCODE,-           ;EXTRACT I/O FUNCTION CODE
          51    20 A3    06         0167    692                  #IRP$S_FCODE,IRP$W_FUNC(R3),R1  ;...
              0092 C5   51    90  016B    693          MOVB    R1,UCB$B_FEX(R5)         ;STORE FUNCTION DISPATCH INDEX
                   51   02    91  0170    694          CMPB    #IO$_SEEK,R1             ;SEEK FUNCTION?
                        08    12  0173    695          BNEQ    30$                      ;IF NEQ - NO
              00BC C5   10    78  0175    696          ASHL    #16,UCB$L_MEDIA(R5),-    ;SHIFT CYLINDER ADDRESS
                    00BC C5       017A    697                  UCB$L_MEDIA(R5)          ;...INTO HIGH WORD
                          AA      017D    698 30$:     BICW    #UCB$M_DIAGBUF-          ;CLEAR DIAGNOSTIC BUFFER PRESENT
                                  017E    699                  !UCB$M_ECC,-             ;...AND ECC CORRECTION MADE FLAGS
                68 A5     03        017E    700                  UCB$W_DEVSTS(R5)         ;...IN DEVICE STATUS WORD
                          07    E1  0181    701          BBC     #IRP$V_DIAGBUF,-         ;IF CLR - NO DIAG BUFFER
                  04 2A A3          0183    702                  IRP$W_STS(R3),FDISPATCH  ;...
                68 A5     02    A8  0186    703          BISW    #UCB$M_DIAGBUF,UCB$W_DEVSTS(R5)  ;SET DIAG BUFFER PRESENT
                                  018A    704
                                  018A    705
                                  018A    706 ;
                                  018A    707 ;       CENTRAL FUNCTION DISPATCH
                                  018A    708 ;
                                  018A    709
                                  018A    710 FDISPATCH:                                ;FUNCTION DISPATCH
            53    58 A5     D0  018A    711          MOVL    UCB$L_IRP(R5),R3         ;GET IRP ADDRESS
                            08    E0  018E    712          BBS     #IRP$V_PHYSIO,-          ;IF SET - PHYSICAL I/O FUNCTION
                  10 2A A3          0190    713                  IRP$W_STS(R3),10$        ;...
                            0B    E0  0193    714          BBS     #UCB$V_VALID,-           ;IF SET - VOLUME SOFTWARE VALID
                0B 64 A5          0195    715                  UCB$W_STS(R5),10$        ;...
          50     0254 8F    3C  0198    716          MOVZWL  #SS$_VOLINV,R0           ;SET VOLUME INVALID STATUS
                   7E A5     B4  019D    717          CLRW    UCB$W_BCNT(R5)           ;SET ZERO BYTES TRANSFERRED
                    017E     31  01A0    718          BRW     FUNCXT                   ;AND RETURN TO CALLER
                                  01A3    719
                00C9 C5     94  01A3    720 10$:     CLRB    UCB$B_DQ_FLAGS(R5)       ;CLEAR LOCAL FLAGS
            53  0092 C5     9A  01A7    721          MOVZBL  UCB$B_FEX(R5),R3         ;GET FUNCTION DISPATCH INDEX
                                  01AC    722          CASE    R3,<-                    ;DISPATCH TO FUNCTION HANDLING ROUTINE
                                  01AC    723                  NOP,-                    ; NO OPERATION
                                  01AC    724                  UNLOAD,-                 ; UNLOAD
                                  01AC    725                  SEEK,-                   ; SEEK
                                  01AC    726                  RECAL,-                  ; RECALIBRATE
                                  01AC    727                  DRVCLR,-                 ; DRIVE CLEAR
                                  01AC    728                  RELEASE,-                ; RELEASE
                                  01AC    729                  OFFSET,-                 ; OFFSET HEADS
                                  01AC    730                  RETCENTER,-              ; RETURN TO CENTER
                                  01AC    731                  PACKACK,-                ; PACKACK
                                  01AC    732                  STARTSPNDL,-             ; START SPINDLE
                                  01AC    733                  WRITECHECK,-             ; WRITE CHECK
                                  01AC    734                  WRITEDATA,-              ; WRITE DATA
                                  01AC    735                  READDATA,-               ; READ DATA
                                  01AC    736                  WRITEHEAD,-              ; WRITE HEADER
                                  01AC    737                  READHEAD,-               ; READ HEADER
                                  01AC    738                  WRITETRACKD,-            ; WRITE TRACK DESCRIPTOR
```

```
                01AC   739                     READTRACKD,-            ; READ TRACK DESCRIPTOR
                01AC   740                     AVAILABLE,-             ; UNIT AVAILABLE
                01AC   741                     >
                01D4   742
                01D4   743 ;
                01D4   744 ; IO$_UNLOAD AND IO$_AVAILABLE INDICATE THE UNIT IS NOT MOUNTED
                01D4   745 ; SO WE CLEAR SOFTWARE VOLUME VALID.  IO$_PACKACK INDICATES THAT
                01D4   746 ; SOFTWARE IS READY TO MOUNT OR ACCESS VOLUE SO WE SET SOFTWARE
                01D4   747 ; VALID.  ON PACKACKS'S WE FOLLOW THIS WITH A GET STATUS AND RESET.
                01D4   748 ; IF THE OBTAINED STATUS INDICATES THAT THE DRIVE IS NOT READY
                01D4   749 ; THEN VOLUME VALID WILL BE CLEARED.
                01D4   750 ;
                01D4   751
                01D4   752 PACKACK:                                     ;PACK ACKNOWLEDGE
          0B    E2    01D4   753              BBSS    #UCB$V_VALID,-          ;SET SOFTWARE VOLUME VALID
0E 64 A5                01D6   754                      UCB$W_STS(R5),NOP       ;...
                01D9   755              EXFUNCL RETRYERR,F_DRVCLR       ;GET STS AND RESET, RETRY ERRORS
          43    11    01E0   756              BRB     NORMAL                  ;SUCCESSFUL - EXIT WITH NORMAL STATUS
                01E2   757
                01E2   758 UNLOAD:                                      ;UNLOAD
                01E2   759 AVAILABLE:                                   ;UNIT AVAILABLE
          0B    E5    01E2   760              BBCC    #UCB$V_VALID,-          ;CLEAR SOFTWARE VALID
00 64 A5                01E4   761                      UCB$W_STS(R5),NOP       ;...
                01E7   762 NOP:                                         ;NO-OP
                01E7   763 RELEASE:                                     ;RELEASE PORT (NOP)
                01E7   764 OFFSET:                                      ;OFFSET HEADS (NOP)
                01E7   765 RETCENTER:                                   ;RETURN TO CENTERLINE (NOP)
                01E7   766 STARTSPNDL:                                  ;START SPINDLE (NOP)
                01E7   767 WRITETRACKD:                                 ;WRITE TRACK DESCRIPTOR (NOP)
                01E7   768 READTRACKD:                                  ;READ TRACK DESCRIPTOR (NOP)
                01E7   769              EXFUNCL RETRYERR,F_NOP          ;EXECUTE A HARDWARE NOP, RETRY ERRORS
          35    11    01EE   770              BRB     NORMAL                  ;SUCCESSFUL - EXIT WITH NORMAL STATUS
                01F0   771
                01F0   772 SEEK:                                        ;SEEK
                01F0   773 RECAL:                                       ;RECALIBRATE
                01F0   774 DRVCLR:                                      ;DRIVE CLEAR (GET STATUS & RESET)
                01F0   775 WRITEHEAD:                                   ;WRITE HEADERS (AND DATA)
                01F0   776              EXFUNCL RETRYERR                ;EXECUTE FUNCTION - RETRY IF FAILURE
          2F    11    01F4   777              BRB     NORMAL                  ;SUCCESSFUL - EXIT WITH NORMAL STATUS
                01F6   778
                01F6   779 WRITECHECK:                                  ;WRITE CHECK
                01F6   780 READHEAD:                                    ;READ HEADER
    4000  8F    AA    01F6   781              BICW    #IO$M_DATACHECK,-       ;CLEAR DATA CHECK REQUEST-
    009A  C5          01FA   782                      UCB$W_FUNC(R5)          ;TO PREVENT EXTRA WRITE CHECK
                01FD   783
                01FD   784 WRITEDATA:                                   ;WRITE DATA
                01FD   785 READDATA:                                    ;READ DATA
    00BE  C5    B1    01FD   786              CMPW    UCB$L_MEDIA+2(R5),-     ;NEW CYLINDER?
    00F4  C5          0201   787                      UCB$L_DQ_CURDA+2(R5)    ;...
          0F    12    0204   788              BNEQ    20$                     ;BRANCH IF SO
                0206   789              BDRVTYP RB80,TRANSFER           ;BRANCH IF RB80
    00BD  C5    91    020C   790              CMPB    UCB$L_MEDIA+1(R5),-     ;OR NEW TRACK? (MUST DO SEEK TO
    00F3  C5          0210   791                      UCB$L_DQ_CURDA+1(R5)    ;...SELECT HEAD ON RB02)
          0C    13    0213   792              BEQL    TRANSFER                ;BRANCH IF NO SEEK REQUIRED
                0215   793 20$:    EXFUNCL RETRYERR,F_SEEK         ;EXECUTE EXPLICIT SEEK - RETRY IF ERROR
53  0092  C5    9A    021C   794              MOVZBL  UCB$B_FUNC(R5),R3       ;RESTORE FUNCTION DISPATCH INDEX
                0221   795
```

```
                     0221   796 ;
                     0221   797 ;       DRIVE HAS BEEN POSITIONED -- NOW EXECUTE THE TRANSFER
                     0221   798 ;
                     0221   799
                     0221   800 TRANSFER:
                     0221   801         EXFUNCL CHECKECC                ;EXECUTE TRANSFER FUNCTION
                     0225   802
                     0225   803
                     0225   804 ;
                     0225   805 ;       OPERATON COMPLETION
                     0225   806 ;
                     0225   807
                     0225   808 NORMAL:                                 ;SUCCESSFUL OPERATION COMPLETE
50   0639 BF    3C   0225   809         MOVZWL  #SS$_WASECC,R0          ;ASSUME CORRECTED ECC ERROR
        00    E0   022A   810         BBS     #UCB$V_ECC,-            ;BRANCH IF CORRECTED ECC
    03 68 A5         022C   811                 UCB$W_DEVSTS(R5),10$    ;...ERROR OCCURED
     50    01    3C   022F   812         MOVZWL  #SS$_NORMAL,R0          ;SET NORMAL COMPLETION STATUS
        00EC    31   0232   813 10$:     BRW     FUNCXT                  ;FUNCTION EXIT
                     0235   814
```

```
                          0235          .SBTTL  RETRIABLE ERROR ANALYSIS
                          0235  ;
                          0235  ;       A RETRIALABLE ERROR HAS OCCURED ON A TRANSFER
                          0235  ;       CHECK TO SEE IF ECC CORRECTION CAN BE APPLIED
                          0235  ;
                          0235  ;       R1      - CSR AT TIME OF ERROR
                          0235  ;       R2      - MPR OF GET STATUS FOLLOWING ERROR
                          0235  ;
                          0235  CHECKECC:
                  01  ED  0235          CMPZV   #RB_CS_V_FCODE,-        ;WAS THIS A READ DATA OPERATION?
       06   51    03      0237                  #RB_CS_S_FCODE,R1,-     ;...
                          023A                  #<F_READDATA @ -1>      ;...
                  6E  12  023A          BNEQ    RETRYERR                ;BRANCH IF NOT
                          023C          BDRVTYP RB02,RETRYERR           ;BRANCH IF RB02
                      B3  0242          BITW    #RB_CS_M_DE-            ;DRIVE ERROR
                          0243                  !RB_CS_M_NXM-           ;...OR NON EXISTENT MEMORY
                          0243                  !RB_CS_M_DLT-           ;...OR DATA LATE
                          0243                  !RB_CS_M_OPI,-          ;...OR OPERATION INCOMPLETE (HDR CRC)
       51  7400   8F      0243                  R1                      ;...?
                  61  12  0247          BNEQ    RETRYERR                ;BRANCH IF SO
         5D 51    0B  E1  0249          BBC     #RB_CS_V_DCK,R1,RETRYERR;BRANCH IF NOT A DATACHECK
                  14  ED  024D          CMPZV   #RB_CS_V_ECS,-          ;COMPARE ECC STATUS BITS (START)
                  02      024F                  #RB_CS_S_ECS,-          ;...(SIZE)
            03    51      0250                  R1,-                    ;...(FROM)
                          0252                  #^B11                   ;...TO BINARY 11 (BOTH SET)
                  56  12  0252          BNEQ    RETRYERR                ;BRANCH IF NOT CORRECTABLE
                          0254
                          0254  ;
                          0254  ;       THIS IS A CORRECTABLE ECC ERROR -- IF IT IS A SINGLE BIT
                          0254  ;       ERROR THEN APPLY THE ECC CORRECTION.  IF IT IS A MULTIPLE
                          0254  ;       BIT ERROR THEN FINISH PROCESSING THE GOOD BLOCKS IN FRONT
                          0254  ;       OF THE ERROR, REREAD THE ERROR BLOCK UNTIL THE ERROR IS
                          0254  ;       CORRECTED OR THE RETRY COUNT IS ZERO.  IF THE RETRY COUNT
                          0254  ;       REACHES ZERO THEN APPLY CORRECTION.
                          0254  ;
          8000 8F     AA  0254          BICW    #RB_CS_M_CE,-           ;CLEAR COMBINED ERROR IN
          00CC C5         0258                  UCB$L_DQ_CS(R5)         ;...CASE WE CONTINUE
       50   7E A5     3C  025B          MOVZWL  UCB$W_BCNT(R5),R0       ;FETCH ORIGINAL XFER COUNT (AS CORRECTED
                          025F                                          ;...BY RETREG BUT INCLUDING ECC BLOCK)
       50   00D4 C5   C0  025F          ADDL    UCB$L_DQ_BC(R5),R0      ;COMPUTE BYTES TRANSFERED
    50  00000200 8F   C2  0264          SUBL    #^X200,R0               ;BACKUP TO LAST GOOD BLOCK
                 3D   19  026B          BLSS    RETRYERR                ;NEGATIVE, SOMETHING WRONG, TRY AGAIN
       50   01FF 8F   B3  026D          BITW    #^X1FF,R0               ;WHOLE BLOCKS TRANSFERED?
                 36   12  0272          BNEQ    RETRYERR                ;NO, SOMETHING WRONG, TRY AGAIN
            7E   52   7D  0274          MOVQ    R2,-(SP)                ;SAVE WORK REGISTERS
  52  00C6 C5    0B   00  EA  0277      FFS     #0,#11,UCB$W_EC2(R5),R2 ;FIND THE FIRST ERROR BIT SET IN THE
                          027E                                          ;...ECC PATERN REGISTER
       53   0A   52   C3  027E          SUBL3   R2,#10,R3               ;GET THE NUMBER OF SET ERROR BITS IN
                          0282                                          ;...THE REMAINDER OF THE PATTERN
                 09   15  0282          BLEQ    10$                     ;BRANCH IF NO OTHER BITS SET
                 52   D6  0284          INCL    R2                      ;POINT TO NEXT BIT IN PATERN
  52  00C6 C5    53   52  EF  0286      EXTZV   R2,R3,UCB$W_EC2(R5),R2  ;IS THERE MORE THAN ONE ERROR BIT SET?
                 0C   BA  028D  10$:    POPR    #^M<R3,R2>              ;RESTORE WORK REGISTERS WITHOUT
                          028F                                          ;...AFFECTING FLAGS
                 10   1B  028F          BLEQU   APPLY_ECC               ;IF ONLY ONE ERROR BIT SET, THEN APPLY
                          0291                                          ;...ECC CORRECTION
                 04   88  0291          BISB    #UCB$M_DQ_ECC_DEFER,-   ;SIGNAL ECC CORRECTION DEFERRED
```

```
                00C9 C5          0293   873                     UCB$B_DQ_FLAGS(R5)      ;
                                 0296   874
      7E A5     0200 8F    A2    0296   875             SUBW    #^X200,UCB$W_BCNT(R5)   ;SHORTEN XFER BY ONE PAGE TO EXCLUDE ECC
                                 029C   876                                             ;...BLOCK FROM NUMBER OF BYTES
                                 029C   877                                             ;...TRANSFERRED
                      0C   13    029C   878             BEQL    RETRYERR                ;BRANCH IF NO GOOD DATA WAS TRANSFERRED
                                 029E   879                                             ;...AND ATTEMPT TO RETRY
                    034D   31    029E   880             BRW     WRITECHK                ;OTHERWISE, BRANCH AND PERFORM A
                                 02A1   881                                             ;...WRITECHECK
                                 02A1   882
                                 02A1   883 APPLY_ECC:
          00000000'GF    16    02A1   884             JSB     G^IOC$APPLYECC          ;APPLY ECC CORRECTION
                    035B   31    02A7   885             BRW     UPDATE                  ;CONTINUE TRANSFER BUT SUPPRESS
                                 02AA   886                                             ;...WRITECHECK
                                 02AA   887
                                 02AA   888 ;
                                 02AA   889 ;       A RETRIABLE ERROR HAS OCCURED
                                 02AA   890 ;
                                 02AA   891 ;       R1      - CSR AT TIME OF ERROR
                                 02AA   892 ;       R2      - MPR OF GET STATUS FOLLOWING ERROR
                                 02AA   893 ;
                                 02AA   894
                                 02AA   895 RETRYERR:
                00C0 C5    97    02AA   896             DECB    UCB$B_ERTCNT(R5)        ;ANY RETRIES LEFT?
                      08   14    02AE   897             BGTR    RESETDRIVE              ;IF GTR - YES
                      02   E4    02B0   898             BBSC    #UCB$V_DQ_ECC_DEFER,-   ;CORRECT THE ERROR WITH ECC IF WE CAN
                00C9 C5          02B2   899                     UCB$B_DQ_FLAGS(R5),-    ;
                      EB         02B5   900                     APPLY_ECC               ;
                      0A   11    02B6   901             BRB     FATALERR                ;OTHERWISE, FATAL ERROR
                                 02B8   902
                                 02B8   903 ;
                                 02B8   904 ; ATTEMPT TO RESET STUBBORN DRIVE ERRORS BEFORE EXECUTING THE FUNCTION AGAIN
                                 02B8   905 ;
                                 02B8   906
                                 02B8   907 RESETDRIVE:
                                 02B8   908             EXFUNCL RETRYERR,F_RECAL        ;RECALIBRATE THE DRIVE
                    FEC8   31    02BF   909             BRW     FDISPATCH               ;RETRY FUNCTION
                                 02C2   910
```

```
                                                L 13
                     02C2   912          .SBTTL  FATAL ERROR ANALYSIS
                     02C2   913
                     02C2   914 ;
                     02C2   915 ;        A FATAL ERROR HAS OCCURED
                     02C2   916 ;
                     02C2   917 ;        R1      - CSR AT TIME OF ERROR OR
                     02C2   918 ;        R2      - MPR OF GET STATUS FOLLOWING ERROR
                     02C2   919 ;
                     02C2   920 FATALERR:                                ;UNRECOVERABLE ERROR
                     02C2   921          ASSUME  RB_MP_V_WL EQ RB_MP_V_WTP       ;ASSUME RB02 AND RB80 USE
                     02C2   922                                          ;SAME BIT FOR WRITE LOCK
                     02C2   923
  13 52    0D   E1   02C2   924          BBC     #RB_MP_V_WL,R2,30$      ;BRANCH IF DRIVE IS NOT WRITELOCKED
50   025C 8F   3C   02C6   925          MOVZWL  #SS$_WRITLCK,R0         ;ASSUME WRITELOCK ERROR STATUS
     0B   91   02CB   926          CMPB    #CDF_WRITEDATA,-        ;WAS THIS A WRITE DATA OPERATION?
     0093 C5           02CD   927                  UCB$B_CEX(R5)           ;...
          4F   13   02D0   928          BEQL    FUNCXT                  ;BRANCH IF SO
     0D   91   02D2   929          CMPB    #CDF_WRITEHEAD,-        ;WAS THIS A WRITE HEADER OPERATION?
     0093 C5           02D4   930                  UCB$B_CEX(R5)           ;...
          48   13   02D7   931          BEQL    FUNCXT                  ;BRANCH IF SO
                     02D9   932
50   0254 8F   3C   02D9   933 30$:     MOVZWL  #SS$_VOLINV,R0          ;ASSUME VOLUME INVALID
                     02DE   934          BDRVTYP RB02,50$                ;BRANCH IF RB02
                     02E4   935
                     02E4   936 ;
                     02E4   937 ;        RB80 ERROR ANALYSIS
                     02E4   938 ;
          11   11   02E4   939          BRB     70$                     ;CONTINUE IN COMMON
                     02E6   940
                     02E6   941
                     02E6   942 ;
                     02E6   943 ;        RB02 ERROR ANALYSIS
                     02E6   944 ;
                     02E6   945
  37 52    09   E0   02E6   946 50$:     BBS     #RB_MP_V_VC,R2,FUNCXT   ;IF SET - VOLUME INVALID
50   025C 8F   3C   02EA   947          MOVZWL  #SS$_WRITLCK,R0         ;ASSUME WRITE LOCK ERROR STATUS
  04 52    0D   E1   02EF   948          BBC     #RB_MP_V_WL,R2,70$      ;IF CLR - VOLUME NOT WRITE LOCKED
  2A 52    0A   F0   02F3   949          BBS     #RB_MP_V_WGE,R2,FUNCXT  ;IF SET - WRITE GATE ERROR
                     02F7   950
50   005C 8F   3C   02F7   951 70$:     MOVZWL  #SS$_DATACHECK,R0       ;ASSUME DATA CHECK ERROR STATUS
          01   E1   02FC   952          BBC     #UCB$V_DQ_DIP,-         ;BRANCH IF NO DATA CHECK IN PROGRESS
  08 00C9 C5           02FE   953                  UCB$B_DQ_FLAGS(R5),80$  ;...
  04 51    0A   E1   0302   954          BBC     #RB_CS_V_OPI,R1,80$     ;DATA CHECK INDICATED BY OPI AND
  17 51    0B   E0   0306   955          BBS     #RB_CS_V_DCK,R1,FUNCXT  ;... DATA CHECK SET
                     030A   956
50   01F4 8F   3C   030A   957 80$:     MOVZWL  #SS$_PARITY,R0          ;ASSUME PARITY ERROR STATUS
  0E 51    0B   E0   030F   958          BBS     #RB_CS_V_DCK,R1,FUNCXT  ;IF SET - CRC ERROR
                     0313   959
50   008C 8F   3C   0313   960 90$:     MOVZWL  #SS$_DRVERR,R0          ;ASSUME DRIVE ERROR STATUS
  05 51    0E   E0   0318   961          BBS     #RB_CS_V_DE,R1,FUNCXT   ;IF SET - DRIVE ERROR
                     031C   962
50   0054 8F   3C   031C   963          MOVZWL  #SS$_CTRLERR,R0         ;ASSUME CONTROLLER ERROR STATUS
                     0321   964
```

```
                         0321   966          .SBTTL  FUNCTION COMPLETION
                         0321   967
                         0321   968 ;
                         0321   969 ;       HERE WHEN THE FUNCTION IS COMPLETE (OR ABORTED) TO
                         0321   970 ;       EXIT FROM DRIVER WITH REQUEST COMPLETE
                         0321   971 ;
                         0321   972
                         0321   973 FUNCXT:                                ;FUNCTION EXIT
            50    DD     0321   974          PUSHL   R0                      ;SAVE FINAL REQUEST STATUS
 00000000'GF    16     0323   975          JSB     G^IOC$DIAGBUFILL        ;FILL DIAGNOSTIC BUFFER IF PRESENT
0092 C5     0A    91     0329   976          CMPB    #CDF_WRITECHECK,UCB$B_FEX(R5)  ;DRIVE RELATED FUNCTION?
            19    1A     032E   977          BGTRU   50$                     ;IF GTRU - YES
  53    58 A5    D0     0330   978          MOVL    UCB$L_IRP(R5),R3        ;RETRIEVE ADDRESS OF IRP
      0A 6E     E8     0334   979          BLBS    (SP),10$                ;BRANCH IF XFER SUCCESSFUL
                         0337   980 ;
                         0337   981 ;THE TRANSFER ENDED IN AN ERROR -- COMPUTE BYTES SUCCESSFULLY TRANSFERRED.
                         0337   982 ;THE BYTE-COUNT-REMAINING REPORTED BY THE DRIVE CANNOT BE TRUSTED IF A
                         0337   983 ;DRIVE ERROR OCCURED.
                         0337   984 ;
  06 51    0E    E0     0337   985          BBS     #RB_CS_V_DE,R1,10$      ;IGNORE FINAL SEGMENT IF DRIVE ERROR
       7E A5    A2     033B   986          SUBW    UCB$W_BCNT(R5),-        ;UPDATE BCR WITH PARTIAL XFER COUNT
     00C0 C5          033E   987                  UCB$W_BCR(R5)           ;...FROM FINAL SEGMENT
     00C0 C5    A3    0341   988 10$:      SUBW3   UCB$W_BCR(R5),-         ;CALCULATE BYTES TRANSFERRED
02 AE    32 A3         0345   989                  IRP$W_BCNT(R3),2(SP)    ;...
                         0349   990
                         0349   991 50$:      RELCHAN                         ;RELEASE CHANNEL IF OWNED
            51    D4     034F   992          CLRL    R1                      ;CLEAR SECOND STATUS LONGWORD
            50 8ED0      0351   993          POPL    R0                      ;RETRIEVE FINAL REQUEST STATUS
                         0354   994          REQCOM                          ;COMPLETE REQUEST
                         035A   995
```

```
                        .SBTTL  HARDWARE FUNCTION DISPATCH
                ;
                ; FEXL - RB730 HARDWARE FUNCTION EXECUTION
                ;
                ; THIS ROUTINE IS CALLED VIA A BSB WITH A BYTE IMMEDIATELY FOLLOWING THAT
                ; SPECIFIES THE ADDRESS OF AN ERROR ROUTINE. ALL DATA IS ASSUMED TO HAVE BEEN
                ; SET UP IN THE UCB BEFORE THE CALL. THE APPROPRIATE PARAMETERS ARE LOADED
                ; INTO DEVICE REGISTERS AND THE FUNCTION IS INITIATED. THE RETURN ADDRESS
                ; IS STORED IN THE UCB AND A WAITFOR INTERRUPT IS EXECUTED. WHEN THE
                ; INTERRUPT OCCURS, CONTROL IS RETURNED TO THE CALLER.
                ;
                ; INPUTS:
                ;
                ;       R3 = FUNCTION TABLE DISPATCH INDEX
                ;       R5 = DEVICE UNIT UCB ADDRESS
                ;
                ;       00(SP) = RETURN ADDRESS OF CALLER
                ;       04(SP) = RETURN ADDRESS OF CALLER'S CALLER
                ;
                ;       IMMEDIATELY FOLLOWING INLINE AT THE CALL SITE IS A BYTE WHICH CONTAINS
                ;       A BRANCH DESTINATION TO AN ERROR RETRY ROUTINE.
                ;
                ; OUTPUTS:
                ;
                ;       THERE ARE FOUR EXITS FROM THIS ROUTINE:
                ;
                ;       1. SPECIAL CONDITION - THIS EXIT IS TAKEN IF A POWER FAILURE OCCURS
                ;             OR THE OPERATION TIMES OUT. IT IS A JUMP TO THE APPROPRIATE
                ;             ERROR ROUTINE.  NO DEVICE REGISTERS ARE SAVED.
                ;
                ;       2. FATAL ERROR - THIS EXIT IS TAKEN IF A FATAL CONTROLLER OR DRIVE
                ;             ERROR OCCURS OR IF ANY ERROR OCCURS AND ERROR RETRY IS EITHER
                ;             INHIBITED OR EXHAUSTED. IT IS A JUMP TO THE FATAL ERROR EXIT
                ;             ROUTINE.  ALL DEVICE REGISTERS ARE SAVED.
                ;
                ;       3. RETRIABLE ERROR - THIS EXIT IS TAKEN IF A RETRIABLE CONTROLLER
                ;             OR DRIVE ERROR OCCURS AND ERROR RETRY IS NEITHER INHIBITED
                ;             NOR EXHAUSTED. IT CONSISTS OF TAKING THE ERROR BRANCH EXIT
                ;             SPECIFIED AT THE CALL SITE.  ALL DEVICE REGISTERS ARE SAVED.
                ;
                ;       4. SUCCESSFUL OPERATION - THIS EXIT IS TAKEN IF NO ERRORS OCCUR
                ;             DURING THE OPERATION. IT CONSISTS OF A RETURN INLINE.
                ;             ONLY THE CSR IS SAVED.
                ;
                ;       IN ALL CASES IF AN ERROR OCCURS, AN ATTEMPT IS MADE TO LOG THE ERROR.
                ;
                ;
```

```
                          035A  1046 FEXL:                                                ;FUNCTION EXECUTOR
      009C C5 8ED0        035A  1047          POPL    UCB$L_DPC(R5)                      ;SAVE DRIVER PC VALUE
0093  C5    53   90        035F  1048          MOVB    R3,UCB$B_CEX(R5)                   ;SAVE CASE INDEX
  50    24 A5   D0        0364  1049          MOVL    UCB$L_CRB(R5),R0                   ;GET ADDRESS OF PRIMARY CRB
  51    2C A0   D0        0368  1050          MOVL    CRB$L_INTD+VEC$L_IDB(R0),R1  ;GET ADDRESS OF IDB
  04 A1       55   D1        036C  1051          CMPL    R5,IDB$L_OWNER(R1)                 ;DOES THIS PROCESS OWN CHANNEL?
              05   12        0370  1052          BNEQ    10$                                ;IF NEQ - NO
      54      61   D0        0372  1053          MOVL    IDB$L_CSR(R1),R4                   ;SET ASSIGNED CHANNEL CSR ADDRESS
              06   11        0375  1054          BRB     20$                                ;
                          0377  1055 10$:     REQPCHAN                                   ;REQUEST CHANNEL (RETURNS R4 = CSR ADR)
                          037D  1056                                                     ;
                          037D  1057 20$:     CASE    R3,<-                              ;DISPATCH TO PROPER FUNCTION ROUTINE
                          037D  1058                  IMMED,-                            ;NO OPERATION
                          037D  1059                  IMMED,-                            ;UNLOAD VOLUME (NOP)
                          037D  1060                  POSIT,-                            ;SEEK CYLINDER
                          037D  1061                  RECALB,-                           ;RECALIBRATE
                          037D  1062                  DRCLR,-                            ;DRIVE CLEAR (GET STATUS & RESET)
                          037D  1063                  IMMED,-                            ;RELEASE DRIVE (NOP)
                          037D  1064                  IMMED,-                            ;OFFSET HEADS (NOP)
                          037D  1065                  IMMED,-                            ;RETURN TO CENTERLINE (NOP)
                          037D  1066                  DRCLR,-                            ;PACK ACKNOWLEDGE
                          037D  1067                  IMMED,-                            ;START SPINDLE (NOP)
                          037D  1068                  >                                  ;
          0142    31      0395  1069          BRW     XFER                               ;TRANSFER FUNCTION
```

```
                     0398  1071          .SBTTL  IMEDIATE FUNCTION EXECUTION
                     0398  1072 ;
                     0398  1073 ; IMMEDIATE FUNCTION EXECUTION
                     0398  1074 ;
                     0398  1075 ;       FUNCTIONS INCLUDE:
                     0398  1076 ;
                     0398  1077 ;               NO OPERATION,
                     0398  1078 ;               DRIVE CLEAR, AND
                     0398  1079 ;               PACK ACKNOWLEDGE
                     0398  1080 ;
                     0398  1081 ; INPUTS:
                     0398  1082 ;       R3      - CASE INDEX
                     0398  1083 ;       R4      - CSR ADDRESS
                     0398  1084 ;       R5      - UCB ADDRESS
                     0398  1085 ;
                     0398  1086 ; FUNCTIONAL DESRIPTION:
                     0398  1087 ;
                     0398  1088 ; INTERRUPTS ARE LOCKED OUT, THE APPROPRIATE FUNCTION IS INITIATED WITH
                     0398  1089 ; INTERRUPT ENABLE, AND A WAITFOR INTERRUPT AND KEEP CHANNEL IS EXECUTED.
                     0398  1090 ;
                     0398  1091 ; THE RESET COMMAND DOES NOT AFFECT AN RB0 SO A RECALIBRATE, WHICH CLEARS
                     0398  1092 ; ERRORS, IS PERFORMED INSTEAD
                     0398  1093 ;
                     0398  1094
                     0398  1095 DRCLR:                                ;GET STATUS AND RESET
                     0398  1096          BDRVTYP RB02,10$                ;BRANCH IF RB02
          53   03   9A 039E  1097          MOVZBL  #CDF_RECAL,R3           ;SET FUNCTION AS RECALIBRATE
0093 C5   53   90   03A1  1098          MOVB    R3,UCB$B_CEX(R5)        ;SAVE CASE INDEX
          3E   11   03A6  1099          BRB     RECALB                  ;AND BRANCH TO EXECUTION
                     03A8  1100
                  D0 03A8  1101 10$:    MOVL    #RB_MP_M_STS-           ;GET STATUS AND
                     03A9  1102                  !RB_MP_M_RST-           ;...RESET DRIVE
                     03A9  1103                  !RB_MP_M_MRK,-          ;...INDICATE GET STATUS COMMAND PRESENT
          50   0B        03A9  1104                  R0                      ;...IN R0
               03   11   03AB  1105          BRB     EX_IMED                 ;EXECUTE THE FUNCTION
                     03AD  1106
                     03AD  1107 IMMED:                               ;GET STATUS
                  D0 03AD  1108          MOVL    #RB_MP_M_STS-           ;GET STATUS AND
                     03AE  1109                  !RB_MP_M_MRK,-          ;...INDICATE GET STATUS COMMAND PRESENT
          50   03        03AE  1110                  R0                      ;...IN R0
                     03B0  1111
                     03B0  1112 EX_IMED:                             ;EXECUTE IMMEDIATE FUNCTION
     52    FC83 CF43   D0 03B0  1113          MOVL    FTAB[R3],R2             ;FETCH FUNCTION CODE AND MODIFIERS
52   02   08    54 A5   F0 03B6  1114          INSV    UCB$W_UNIT(R5),#8,#2,R2 ;MERGE UNIT NUMBER
                     03BC  1115          SAVIPL                          ;SAVE CURRENT IPL ON STACK
                     03BF  1116          CKPWR   10$                     ;RAISE IPL AND CHECK FOR POWERFAIL
          10 A4    50   D0 03CC  1117          MOVL    R0,RB_MP(R4)            ;PREPARE FOR GETSTATUS OR RESET
             64    52   D0 03D0  1118          MOVL    R2,RB_CS(R4)            ;INITIATE FUNCTION
                     03D3  1119          WFIKPCH RETREG,#10              ;WAITFOR INTERRUPT
                     03DD  1120          IOFORK                          ;CREATE FORK PROCESS
                02CA 31   03E3  1121 10$:    BRW     RETREG                  ;
                     03E6  1122
```

```
                        03E6  1124          .SBTTL  RECALIBRATE FUNCTION EXECUTION
                        03E6  1125 ;
                        03E6  1126 ; RECALIBRATE FUNCTION EXECUTION
                        03E6  1127 ;
                        03E6  1128 ;       FUNCTIONS INCLUDE:
                        03E6  1129 ;
                        03E6  1130 ;               RECALIBRATE
                        03E6  1131 ;
                        03E6  1132 ; INPUTS:
                        03E6  1133 ;       R3      - CASE INDEX
                        03E6  1134 ;       R4      - DEVICE CSR ADDRESS
                        03E6  1135 ;       R5      - UCB ADDRESS
                        03E6  1136 ;
                        03E6  1137 ; FUNCTIONAL DESRIPTION:
                        03E6  1138 ;
                        03E6  1139 ; FOR AN RB80, A RECALIBRATE IS PERFORMED.  THE HEADS SEEK TO A KNOWN
                        03E6  1140 ; POSITION TO RESET THE DRIVES PHYSICAL POSITIONING LOGIC.  THIS IS
                        03E6  1141 ; SIGNALED BY REQUESTING A SEEK TO -1
                        03E6  1142 ;
                        03E6  1143 ; FOR AN RB02, A READ HEADER IS PERFORMED.  THE RB02 DOES NOT SUPPORT
                        03E6  1144 ; A RECALIBRATE FUNCTION.  HOWEVER, THE SUPPORT MICROCODE FOR THE RB02
                        03E6  1145 ; KEEPS AN INTERNAL RECORD OF THE CURRENT DISK POSITION.  WHENEVER A
                        03E6  1146 ; READ HEADER IS PERFORMED, IT UPDATES THAT POSITION WITH THE VALUE
                        03E6  1147 ; FOUND IN THE HEADER.
                        03E6  1148 ;
                        03E6  1149
                        03E6  1150 RECALB:                                      ;RECALIBRATE FUNCTION
      00F2 C5   01   CE 03E6  1151          MNEGL   #1,UCB$L_DQ_CURDA(R5)     ;DISABLE SEEK OPTIMIZATION
                        03EB  1152          BDRVTYP RB02,50$                  ;BRANCH IF RB02
                        03F1  1153          GETUNIT                           ;GET UNIT NUMBER IN R2
                        03F9  1154          DSBINT  UCB$B_DIPL(R5)            ;SAVE IPL AND LOCK OUT DEVICE INTERUPTS
64    52    FC33 CF43   C9  0400  1155          BISL3   FTAB[R3],R2,RB_CS(R4)     ;LOAD CSR (EXECUTION SUPPRESED)
            64   01     D3  0407  1156          BITL    #RB_CS_M_DRDY,RB_CS(R4)   ;IS DRIVE READY?
                 04     12  040A  1157          BNEQ    10$                       ;BRANCH IF SO
          1C A4   01     CE  040C  1158          MNEGL   #1,RB_CMD(R4)             ;INITIALIZE ENTIRE SUBSYSTEM
          0C A4   01     CE  0410  1159 10$:     MNEGL   #1,RB_DA(R4)              ;LOAD -1 IN DISK ADDRESS REGISTER
                 2E     11  0414  1160          BRB     SEEKI                     ;INITIATE SEEK
                            0416  1161
            53   0E     9A  0416  1162 50$:     MOVZBL  #CDF_READHEAD,R3          ;SET FUNCTION AS READ HEADER
        0093 C5   53     90  0419  1163          MOVB    R3,UCB$B_CEX(R5)          ;SAVE CASE INDEX
                 00B9   31  041E  1164          BRW     XFER                      ;EXECUTE TRANSFER FUNCTION
                            0421  1165
```

```
                     1167          .SBTTL  POSITIONING FUNCTION EXECUTION
                     1168 ;
                     1169 ; POSITIONING FUNCTION EXECUTION
                     1170 ;
                     1171 ;       FUNCTIONS INCLUDE:
                     1172 ;
                     1173 ;               SEEK CYLINDER
                     1174 ;
                     1175 ; INPUTS:
                     1176 ;       R3      - CASE INDEX
                     1177 ;       R4      - DEVICE CSR ADDRESS
                     1178 ;       R5      - UCB ADDRESS
                     1179 ;
                     1180 ; FUNCTIONAL DESRIPTION:
                     1181 ;
                     1182 ; THE CYLINDER ADDRESS IS LOADED INTO THE DISK ADDRESS REGISTER.
                     1183 ; INTERRUPTS ARE LOCKED OUT, AND THE SEEK FUNCTION IS INITIATED.
                     1184 ; WITH INTERRUPT ENABLE.  THE UNIT MUST BE SELECTED BEFORE LOADING
                     1185 ; THE CYLINDER ADDRESS (SO UCODE KNOWS WHETHER ITS AN RB80 OR RB02).
                     1186 ;
                     1187 ; WHEN THE FIRST INTERRUPT IS RECEIVED THE CHANNEL IS RELEASED (MUST
                     1188 ; OCCUR AT FORK LEVEL) AND THE COMPLETION INTERRUPT IS WAITED FOR.
                     1189 ; THE SEEK MAY COMPLETE WHILE AT FORK LEVEL SO A FLAG IS USED TO
                     1190 ; SYNCRONIZE THE OPERATION.
                     1191 ;
                     1192 ;
                     1193
                     1194 POSIT:                                        ;POSITIONING FUNCTION
                     1195          GETUNIT                             ;GET UNIT NUMBER IN R2
                     1196          DSBINT  UCB$B_DIPL(R5)              ;SAVE IPL AND LOCK OUT DEVICE INTERUPTS
64   52   FC03 CF43   C9   0430    1197          BISL3   FTAB[R3],R2,RB_CS(R4)    ;LOAD CSR (EXECUTION SUPPRESED)
   0C A4    00BC C5   D0   0437    1198          MOVL    UCB$L_MEDIA(R5),RB_DA(R4) ;LOAD CYLINDER ADDRESS IN DAR
            00BC C5   D0   043D    1199          MOVL    UCB$L_MEDIA(R5),-   ;REMEMBER CURRENT DISK ADDRESS
            00F2 C5         0441    1200                  UCB$L_DQ_CURDA(R5)  ;...FOR SEEK OPTIMIZATION
                            0444    1201
                            0444    1202 SEEKI:                             ;SEEK INITIATE
                            0444    1203          CKPWR   25$               ;DISABLE INTERRUPTS, CHECK POWER
                            0451    1204          INITIATE                  ;INITIATE THE FUNCTION
                      01    88  0458    1205          BISB    #UCB$M_DQ_SIP,-   ;SIGNAL SEEK IN PROGRESS
            00C9 C5         045A    1206                  UCB$B_DQ_FLAGS(R5)  ;...
                            045D    1207          BDRVTYP RB02,TO$          ;BRANCH IF RB02
                            0463    1208 ;
                            0463    1209 ; RB80'S INITIATE SEEKS VERY QUICKLY (APPROX 30 USECS).  CONSEQUENTLY
                            0463    1210 ; WE WAIT FOR THE SEEK TO INITIATE IN A LOOP, THEN CLEAR THE INITIATION
                            0463    1211 ; INTERRUPT AND WAIT FOR THE COMPLETION INTERRUPT.
                            0463    1212 ;
                            0463    1213
                            0463    1214          TIMEWAIT #3,#RB_CS_M_CRDY,-   ;WAIT FOR CONTROLLER READY
                            0463    1215                  RB_CS(R4),[         ;... 3*10 MICS
                         0C 50  E9  048A    1216          BLBC    R0,10$        ;BRANCH IF CONTROLLER STILL NOT READY
64   000F0000 8F    CA  048D    1217          BICL    #RB_CS_M_ATN,RB_CS(R4)  ;CLEAR INTERRUPT REQUEST FROM INITIATE
                            0494    1218          ENBINT                    ;DROP IPL AND CLEANUP STACK
                      16    11  0497    1219          BRB     20$           ;
                            0499    1220 ;
                            0499    1221 ; RB02'S CAN TAKE UPTO A FULL SECTOR TIME TO INITIATE AN INTERRUPT.
                            0499    1222 ; CONSEQUENTLY WE TAKE TWO INTERUPTS, ONE FOR SEEK INITIATE, THE OTHER
                            0499    1223 ; FOR SEEK COMPLETION
```

```
                     0499  1224 ;
                     0499  1225 10$:    WFIKPCH RETREG,#10                ;WAIT FOR INITIATION INTERRUPT
          0F   E0    04A3  1226         BBS     #RB_CS_V_CE,-             ;BRANCH IF SEEK INITIATE FAILED
28 00CC   C5         04A5  1227                 UCB$L_DQ_CS(R5),40$       ;...
                     04A9  1228         IOFORK                            ;DROP TO FORK IPL
                     04AF  1229 20$:    RELCHAN                           ;RELEASE THE CHANNEL
                     04B5  1230         DSBINT  UCB$B_DIPL(R5)            ;RETURN TO DEVICE IPL
          00   E4    04BC  1231         BBSC    #UCB$V_DQ_SIP,-           ;BRANCH IF SEEK NOT COMPLETED YET
05 00C9   C5         04BE  1232                 UCB$B_DQ_FLAGS(R5),30$    ;...
                     04C2  1233         ENBINT                            ;RESTORE IPL
          10   11    04C5  1234 25$:    BRB     50$                       ;DON'T WAIT FOR A SECOND INTERRUPT
                     04C7  1235 30$:    WFIKPCH RETREG,#10                ;WAIT FOR COMPLETION (CHANNEL RELEASED)
                     04D1  1236 40$:    IOFORK                            ;DROP TO FORK IPL
        01D6   31    04D7  1237 50$:    BRW     RETREG                    ;SEEK COMPLETION
                     04DA  1238
```

```
                                        .SBTTL  TRANSFER FUNCTION EXECUTION
                                ;
                                ; TRANSFER FUNCTION EXECUTION
                                ;
                                ;       FUNCTIONS INCLUDE:
                                ;
                                ;               WRITE CHECK
                                ;               WRITE DATA
                                ;               READ DATA, AND
                                ;               READ HEADER
                                ;
                                ; INPUTS.
                                ;       R3      - CASE INDEX
                                ;       R4      - DEVICE CSR ADDRESS
                                ;       R5      - UCB ADDRESS
                                ;
                                ; FUNCTIONAL DESCRIPTION:
                                ;
                                ; THE TRANSFER PARAMETERS ARE LOADED INTO THE DEVICE REGISTERS, INTERRUPTS
                                ; ARE LOCKED OUT, THE FUNCTION IS INITIATED, AND A WAITFOR INTERRUPT AND
                                ; KEEP CHANNEL IS EXECUTED.
                                ;
                                ; UPON RETURN FROM THE INTERRUPT SERVICE ROUTINE, IF THE TRANSFER IS
                                ; COMPLETE, THE APPROPRIATE EXIT IS TAKEN. IF THE FUNCTION IS NOT COMPLETE
                                ; TRANSFER PARAMETERS ARE UPDATED AND A RETURN TO FDISPATCH IS EXECUTED TO
                                ; RE-ISSUE SEEK AND TRANSFER FUNCTIONS WHILE KEEPING CHANNEL AND UBA
                                ; RESOURCES. IF A DATA CHECK HAS BEEN REQUESTED, IT IS PERFORMED
                                ; BEFORE RETURNING TO FDISPATCH.
                                ;

                                XFER:                                           ;TRANSFER FUNCTION EXECUTION
                                ;
                                ; LOAD UBA MAPS
                                ;
00C0 C5    B0                   MOVW    UCB$W_BCR(R5),-                      ;GET BYTES LEFT TO TRANSFER AND -
   7E A5                                UCB$W_BCNT(R5)                       ;ASSUME ONLY ONE TRANSFER NEEDED
   53  0E  91                   CMPB    #CDF_READHEAD,R3                     ;IS THIS A READ HEADER OPERATION
       32  13                   BEQL    NOMAPS                               ;BRANCH IF SO, DON'T NEED MAPS
                                ;
                                ;
                                ;
                                ; COMPUTE SIZE OF THIS TRANSFER -- MAXIMUM = 1 TRACK
                                ;
 52   44 A5  9A                 MOVZBL  UCB$B_SECTORS(R5),R2                 ;GET SECTORS/SURFACE
52  00BC C5  82                 SUBB    UCB$W_DA(R5),R2                      ;CALCULATE SECTORS LEFT ON SURFACE
52  0100 8F  A4                 MULW    #256,R2                              ;COMPUTE BYTES REMAINING ON SURFACE
                                BDRVTYP RB02,10$                             ;BRANCH IF AN RB02
   52  02  A4                   MULW    #2,R2                                ;RB80 HAS 512 BYTE SECTORS
       09  E1                   BBC     #IO$V_SKPSECINH,-                    ;BRANCH NO SKIP SECTOR INHIBIT
 05 009A C5                             UCB$W_FUNC(R5),10$                   ;...
52  0200 8F  A0                 ADDW    #512,R2                              ;ALLOW ACCESS TO 32ND SECTOR
 52   7E A5  B1                 CMPW    UCB$W_BCNT(R5),R2                    ;ARE ADDITIONAL TRANSFERS REQUIRED?
       04  1B                   BLEQU   20$                                  ;BRANCH IF NOT
 7E A5   52  B0                 MOVW    R2,UCB$W_BCNT(R5)                    ;STORE PARTIAL TRANSFER BYTE COUNT
                                20$:    LOADUBAA                             ;LOAD UNIBUS MAP REGISTERS
```

```
                     0517  1297 ;
                     0517  1298 ; MAPS LOADED (IF NECESSARY) AND BYTE COUNT DETERMINED.
                     0517  1299 ; LOAD BYTE COUNT
                     0517  1300 ;
                     0517  1301
                     0517  1302 NOMAPS:
                     0517  1303         DSBINT  UCB$B_DIPL(R5)          ;SAVE IPL AND LOCK OUT DEVICE INTERUPTS
 52    7E A5    3C   051E  1304         MOVZWL  UCB$W_BCNT(R5),R2       ;FETCH BYTE COUNT
 08 A4    52    CE   0522  1305         MNEGL   R2,RB_BC(R4)            ;SET NEGATIVE BYTE COUNT
                     0526  1306
                     0526  1307
                     0526  1308 ;
                     0526  1309 ; COMPUTE AND LOAD 18 BIT UNIBUS ADDRESS
                     0526  1310 ;
                     0526  1311
 50    7C A5    3C   0526  1312         MOVZWL  UCB$W_BOFF(R5),R0       ;FETCH BYTE OFFSET
 51    24 A5    D0   052A  1313         MOVL    UCB$L_CRB(R5),R1        ;GET CRB ADDRESS
       34 A1    F0   052E  1314         INSV    CRB$L_INTD+VEC$W_MAPREG(R1),-  ;INSERT STARTING MAP REGISTER
50    09    09         0531  1315                 #9,#9,R0                ;...NUMBER IN HIGH NINE BITS
  04 A4    50   D0   0534  1316         MOVL    R0,RB_BA(R4)            ;SET BUFFER ADDRESS
                     0538  1317
                     0538  1318
                     0538  1319 ;
                     0538  1320 ; PERFORM R80 TRACK-TO-TRACK SPIRALLING
                     0538  1321 ;       THE R80 CAN CHANGE HEADS JUST BY LOADING A SEEK COMMAND,
                     0538  1322 ;       AND LOADING THE DAR.  WE TAKE ADVANTAGE OF THIS FEATURE
                     0538  1323 ;       TO REDUCE SEEK TIMES
                     0538  1324 ;
                     0538  1325
                     0538  1326         GETUNIT                         ;GET UNIT NUMBER IN R2
           00BD C5   91   0540  1327         CMPB    UCB$L_MEDIA+1(R5),-     ;COMPARE DESIRED TRACK
           00F3 C5        0544  1328                 UCB$L_DQ_CURDA+1(R5)    ;...TO CURRENT TRACK
                  18   13   0547  1329         BEQL    20$                     ;BRANCH IF ON TRACK
           53     0E   91   0549  1330         CMPB    #CDF_READHEAD,R3        ;IS THIS A READ HEADER OPERATION?
                  13   13   054C  1331         BEQL    20$                     ;BRANCH IF SO, DON'T CHANGE HEADS
  64    52     FAEE CF   C9   054E  1332         BISL3   FTAB+<CDF_SEEK*4>,R2,-  ;SET CONTROLLER TO SEEK MODE
                     0554  1333                 RB_CS(R4)               ;...
    0C A4     00BC C5   D0   0554  1334         MOVL    UCB$L_MEDIA(R5),RB_DA(R4) ;DO A HEAD SELECT
           00BD C5   90   055A  1335         MOVB    UCB$L_MEDIA+1(R5),-     ;UPDATE CURRENT DISK ADDRESS
           00F3 C5        055E  1336                 UCB$L_DQ_CURDA+1(R5)    ;...WITH NEW TRACK
                     0561  1337
                     0561  1338 ;
                     0561  1339 ; EXECUTE THE TRANSFER FUNCTION --
                     0561  1340 ;       NOTE: THE FUNCTION MUST BE SPECIFIED BEFORE LOADING THE DAR
                     0561  1341 ;       BECAUSE THE UCODE MUST KNOW WHETHER THE TRANSFER IS TO AN
                     0561  1342 ;       RB02 OR AN RB80.
                     0561  1343 ;
                     0561  1344
              09     E1   0561  1345 20$:      BBC     #IO$V_SKPSECINH,-       ;BRANCH NO SKIP SECTOR INHIBIT
        07 009A C5        0563  1346                 UCB$W_FUNC(R5),30$      ;...
    52  08400000 8F   C8   0567  1347         BISL    #RB_CS_M_SSEI-          ;INHIBIT SKIP SECTOR ERRORS
                     056E  1348                 !RB_CS_M_ASSI,R2        ;...AND AUTOMATIC SKIP SECTORING
64    52    FAC5 CF43   C9   056E  1349 30$:      BISL3   FTAB[R3],R2,RB_CS(R4)   ;LOAD UNIT NUMBER AND FUNCTION
                     0575  1350         CKPWR   BRW_RETREG              ;DISABLE INTERRUPTS, CHECK POWER
    0C A4    00BC C5   D0   0582  1351         MOVL    UCB$L_MEDIA(R5),RB_DA(R4);SET DESIRED DISK ADDRESS
                     0588  1352         INITIATE                        ;INITIATE THE FUNCTION
                     058F  1353         WFIKPCH RETREG,#10              ;WAITFOR INTERRUPT AND KEEP CHANNEL
```

```
                                                 I 14
                     0599  1354 ;
                     0599  1355 ;THE R80 PRODUCES SPURIOUS ATTENTION BITS ON XFER'S.  UNTIL FIXED WE
                     0599  1356 ;FOLLOW EACH TRANSFER WITH AN EXPLICIT CLEAR OF THE UNIT'S ATTENTION BIT.
                     0599  1357 ;
51    0F    10   78  0599  1358          ASHL    #RB_CS_V_ATN,#^X0F,R1   ;FORM BIT MASK
  50    54 A5   3C  059D  1359          MOVZWL  UCB$W_UNIT(R5),R0       ;FETCH UNIT NUMBER
     50    10   C0  05A1  1360          ADDL    #RB_CS_V_ATN,R0         ;POINT INTO MASK
  00 51    50   E5  05A4  1361          BBCC    R0,R1,50$               ;CLEAR THIS UNIT'S ATTENTION BIT
     64    51   CA  05A8  1362 50$:     BICL    R1,RB_CS(R4)            ;CLEAR THIS UNIT'S BIT IN THE CSR
                     05AB  1363          IOFORK                          ;CREATE FORK PROCESS (RETURN TO ISR)
           0F    E1  05B1  1364          BBC     #RB_CS_V_CE,-           ;BRANCH IF NO ERRORS
  03 00CC C5         05B3  1365                  UCB$L_DQ_CS(R5),RETHDR  ;...
                     05B7  1366 BRW_RETREG:                      ; WORD DISPLACEMENT, UNCONDITIONAL BRANCH
        00F6   31    05B7  1367          BRW     RETREG                  ;RETURN REGISTERS
                     05BA  1368
```

```
                     05BA  1370          .SBTTL  TRANSFER POST PROCESSING
                     05BA  1371
                     05BA  1372 ;
                     05BA  1373 ; PURGE DATAPATH -- NOTE: THE DATAPATH IS NOT PURGED BECAUSE THIS
                     05BA  1374 ; DRIVER IS SPECIFIC TO THE VAX730 PROCESS WHICH DOES NOT REQUIRE
                     05BA  1375 ; DATAPATH PURGING. CONSEQUENTLY THE DATAPATH REGISTER WILL ALWAYS
                     05BA  1376 ; BE ZERO IN ERRLOG AND DIAGNOSTIC BUFFERS.
                     05BA  1377 ;
                     05BA  1378
                     05BA  1379
                     05BA  1380 ;
                     05BA  1381 ; RETURN HEADER INFORMATION FOR READ HEADER FUNCTION --
                     05BA  1382 ; IF AN INTERNAL READY HEADER THEN SIMPLY EXIT.
                     05BA  1383 ;
                     05BA  1384
                     05BA  1385 RETHDR:                                  ;RETURN HEADER INFO
0093 C5   0E   91    05BA  1386          CMPB    #CDF_READHEAD, -        ;WAS THIS A READ HEADER?
                     05BF  1387                  UCB$B_CEX(R5)           ;...
          2D   12    05BF  1388          BNEQ    WRITECHK                ;BRANCH IF NOT
0092 C5   0E   91    05C1  1389          CMPB    #IO$_READHEAD, -        ;INTERNAL READ HEADER?
                     05C6  1390                  UCB$B_FEX(R5)           ;...
          EF   12    05C6  1391          BNEQ    BRW_RETREG              ;BRANCH IF SO
        78 A5  DD    05C8  1392          PUSHL   UCB$L_SVAPTE(R5)        ;SAVE ADDRESS OF PTE
51   00EC C5  9E    05CB  1393          MOVAB   UCB$W_DQ_HDR1(R5),R1    ;SET ADDRESS OF INTERNAL BUFFER
     52   06  D0    05D0  1394          MOVL    #6,R2                   ;SET NUMBER OF BYTES TO MOVE
  7E A5   52  B1    05D3  1395          CMPW    R2,UCB$W_BCNT(R5)       ;ROOM FOR FULL HEADER?
          04  1B    05D7  1396          BLEQU   30$                     ;BRANCH IF SO
  52   7E A5  3C    05D9  1397          MOVZWL  UCB$W_BCNT(R5),R2       ;SET LENGTH OF PARTIAL HEADER
00C0 C5   52  A2    05DD  1398 30$:     SUBW2   R2,UCB$W_BCR(R5)        ;UPDATE BYTE COUNT REMAINING
 00000000'GF  16    05E2  1399          JSB     G^IOC$MOVTOUSER         ;MOVE HEADER TO USER BUFFER
        78 A5 8ED0  05E8  1400          POPL    UCB$L_SVAPTE(R5)        ;RESTORE ADDRESS OF PTE
           C9  11    05EC  1401          BRB     BRW_RETREG              ;TERMINATE FUNCTION
                     05EE  1402
```

```
                     05EE  1404          .SBTTL  DATA CHECK AND PARAMETER UPDATE
                     05EE  1405
                     05EE  1406 ;
                     05EE  1407 ; PERFORM WRITE CHECK, IF REQUESTED
                     05EE  1408 ;
                     05EE  1409
                     05EE  1410 WRITECHK:                            ;WRITECHECK AFTER PARTIAL TRANSFER
          0E   E1    05EE  1411          BBC     #IO$V_DATACHECK,-    ;IF CLR - DATA CHECK NOT REQUESTED
 11 009A C5           05F0  1412                  UCB$W_FUNC(R5),UPDATE ;...
          01   E4    05F4  1413          BBSC    #UCB$V_DQ_DIP,-      ;CLEAR DATA CHECK IN PROGRESS
     00C9 C5           05F6  1414                  UCB$B_DQ_FLAGS(R5),- ;...AND BRANCH IF SET
          0B           05F9  1415                  UPDATE               ;...
          02   88    05FA  1416          BISB    #UCB$M_DQ_DIP,-      ;SET DATA CHECK IN PROGRESS
     00C9 C5           05FC  1417                  UCB$B_DQ_FLAGS(R5)   ;...
     53   0A   9A    05FF  1418          MOVZBL  #IO$_WRITECHECK,R3   ;SET CASE INDEX TO WRITE CHECK
        FF12   31    0602  1419          BRW     NOMAPS               ;BRANCH TO PERFORM WRITE CHECK
                     0605  1420
                     0605  1421
                     0605  1422 ;
                     0605  1423 ; UPDATE BUFFER ADDRESS, CURRENT DISK ADDRESS, AND BYTES REMAINING
                     0605  1424 ; FOR NEXT TRANSFER
                     0605  1425 ;
                     0605  1426
                     0605  1427 UPDATE:                              ;UPDATE TRANSFER PARAMETERS
       50   7E A5   3C    0605  1428          MOVZWL  UCB$W_BCNT(R5),R0    ;FETCH BYTES TRANSFERRED
 00C0 C5    50   A2    0609  1429          SUBW    R0,UCB$W_BCR(R5)     ;UPDATE BYTES REMAINING TO XFER
          A7   13    060E  1430          BEQL    BRW_RETREG           ;BRANCH IF TRANSFER COMPLETE
                     0610  1431
                     0610  1432 ;
                     0610  1433 ; COMPUTE NUMBER OF 512 BYTE BLOCKS TRANSFERED
                     0610  1434 ;
50   50    F9 8F    78    0610  1435 10$:     ASHL    #-7,R0,R0            ;COMPUTE PAGES * 4
     78 A5    50   C0    0615  1436          ADDL    R0,UCB$L_SVAPTE(R5)  ;UPDATE THE ADDRESS OF THE PTE
     78 A5    03   8A    0619  1437          BICB2   #^X3,UCB$L_SVAPTE(R5) ;ROUND DOWN TO FULL PAGES (RL02'S!)
50   50    FF 8F    78    061D  1438          ASHL    #-1,R0,R0            ;CONVERT TO 256 BYTE SECTORS
                     0622  1439          BDRVTYP RB80,15$             ;BRANCH IF RB80
          0F 50    E9    0628  1440          BLBC    R0,20$               ;CHECK FOR ODD SECTOR ADDRESSING
          7D A5    96    062B  1441          INCB    UCB$W_BOFF+1(R5)     ;ADD ^X100 TO BOFF
  7D A5    FE 8F    8A    062E  1442          BICB    #^XFE,UCB$W_BOFF+1(R5) ;MAKE BOFF MODULO ^X200
          05   11    0633  1443          BRB     20$                  ;CONTINUE IN COMMON
                     0635  1444
50   50    FF 8F    78    0635  1445 15$:     ASHL    #-1,R0,R0            ;CONVERT TO 512 BYTE SECTORS
   00BC C5    50   80    063A  1446 20$:     ADDB    R0,UCB$W_DA(R5)      ;UPDATE SECTOR
        00BC C5    91    063F  1447          CMPB    UCB$W_DA(R5),-       ;COMPARE UPDATED SECTOR
          44 A5          0643  1448                  UCB$B_SECTORS(R5)    ;...TO SECTORS PER TRACK
          28   1F    0645  1449          BLSSU   50$                  ;BRANCH IF MORE REMAIN
          0E   12    0647  1450          BNEQ    30$                  ;BRANCH IF PAST LOGICAL END OF TRACK
                     0649  1451          BDRVTYP RB02,30$             ;BRANCH IF RB02
          09   E1    064F  1452          BBC     #IO$V_SKPSECINH,-    ;BRANCH NO SKIP SECTOR INHIBIT
     02 009A C5           0651  1453                  UCB$W_FUNC(R5),30$   ;...
                     0655  1454 ;
                     0655  1455 ;THIS IS AN R80 DRIVE, ON THE LAST LOGICAL SECTOR, AND SKIP SECTOR
                     0655  1456 ;INHIBIT IS SET -- THERE IS ONE PHYSICALLY ACCESSABLE BLOCK REMAINING, SO
                     0655  1457 ;CONTINUE ON THE SAME TRACK
                     0655  1458 ;
          18   11    0655  1459          BRB     50$                  ;ONE MORE SECTOR REMAINS
                     0657  1460
```

```
00BC C5   94   0657  1461 30$:    CLRB    UCB$W_DA(R5)            ;CLEAR SECTOR ADDRESS
00BD C5   96   065B  1462           INCB    UCB$W_DA+1(R5)          ;INCREMENT TRACK
00BD C5   91   065F  1463           CMPB    UCB$W_DA+1(R5),-        ;COMPARE UPDATED TRACK
  45 A5        0663  1464                   UCB$B_TRACKS(R5)        ;...TO TRACKS PER CYLINDER
     08   1F   0665  1465           BLSSU   50$                     ;BRANCH IF MORE REMAIN
00BD C5   94   0667  1466           CLRB    UCB$W_DA+1(R5)          ;RESET DESIRED TRACK (SURFACE) TO 0
00BE C5   B6   066B  1467           INCW    UCB$W_DC(R5)            ;INCREMENT CYLINDER
               066F  1468
   FB18   31   066F  1469 50$:     BRW     FDISPATCH               ;MORE BYTES REMAINING - CONTINUE
               0672  1470
```

```
                        0672  1472          .SBTTL  SPECIAL CONDITION (POWER, TIMEOUT)
                        0672  1473
                        0672  1474 ;
                        0672  1475 ; SPECIAL CONDITION EXIT (POWER FAILURE OR DEVICE TIMEOUT)
                        0672  1476 ;
                        0672  1477
                        0672  1478 SPECOND:
    00F2 C5    01   CE  0672  1479          MNEGL   #1,UCB$L_DQ_CURDA(R5)   ;DISABLE SEEK OPTIMIZATION
               05   E4  0677  1480          BBSC    #UCB$V_POWER,-          ;IF SET - POWER FAILURE
         24 64 A5        0679  1481                  UCB$W_STS(R5),PWRFAIL   ;...ELSE TIMEOUT
                        067C  1482          SETIPL  UCB$B_FIPL(R5)          ;TIMEOUTS ENTER AT DEVICE IPL
     00000000'GF   16   0680  1483          JSB     G^ERL$DEVICTMO          ;LOG DEVICE TIMEOUT
           7E A5   B4   0686  1484          CLRW    UCB$W_BCNT(R5)          ;SET ZERO BYTES TRANSFERED
 64 A5    0040 8F   AA  0689  1485          BICW    #UCB$M_TIMOUT,UCB$W_STS(R5)  ;CLEAR TIMEOUT STATUS
 50       022C 8F   3C  068F  1486          MOVZWL  #SS$_TIMEOUT,R0         ;SET DEVICE TIMEOUT STATUS
          0080 C5   97  0694  1487          DECB    UCB$B_ERTCNT(R5)        ;ANY ERROR RETRIES REMAINING?
               03   15  0698  1488          BLEQ    10$                     ;BRANCH IF NOT
             FC1B   31  069A  1489          BRW     RESETDRIVE              ;RETRY THE FUNCTION
             FC81   31  069D  1490 10$:     BRW     FUNCXT                  ;GIVE UP
                        06A0  1491
                        06A0  1492 PWRFAIL:                                ;POWER FAILURE
       64 A5    20   AA  06A0  1493          BICW    #UCB$M_POWER,UCB$W_STS(R5)  ;CLEAR POWER FAILURE BIT
       53    58 A5   D0  06A4  1494          MOVL    UCB$L_IRP(R5),R3        ;GET ADDRESS OF I/O PACKET
          2C A3   7D  06A8  1495          MOVQ    IRP$L_SVAPTE(R3),-      ;RESTORE TRANSFER PARAMETERS
          78 A5       06AB  1496                  UCB$L_SVAPTE(R5)        ;...
             FA6B   31  06AD  1497          BRW     PREPROCESS              ;RETURN TO PREPROCESS UCB FIELDS
                        06B0  1498
```

```
                                06B0  1500          .SBTTL  HARDWARE FUNCTION EXIT PROCESSING
                                06B0  1501
                                06B0  1502 ;
                                06B0  1503 ; DETERMINE EXIT - SPECIAL CONDITION, FATAL ERROR, RETRIABLE ERROR, OR SUCCESS
                                06B0  1504 ;
                                06B0  1505
                             B3 06B0  1506 RETREG: BITW    #UCB$M_POWER!-          ;POWER FAIL
                                06B1  1507                 UCB$M_TIMOUT,-          ;...OR DEVICE TIMEOUT
                64 A5    0060 8F      06B1  1508                 UCB$W_STS(R5)           ;...IN STATUS WORD?
                               BA 12  06B6  1509          BNEQ    SPECOND                 ;BRANCH IF SO -- SPECIAL CONDITION
                   51    00CC C5  D0  06B8  1510          MOVL    UCB$L_DQ_CS(R5),R1      ;FETCH CSR
                     7B 51      0F  E1  06BD  1511          BBC     #RB_CS_V_CE,R1,SUCCESS  ;BRANCH IF NO ERRORS
                   00F2 C5      01  CE  06C1  1512          MNEGL   #1,UCB$L_DQ_CURDA(R5)   ;DISABLE SEEK OPTIMIZATION
                        00D4 C5     A0  06C6  1513          ADDW    UCB$L_DQ_BCT(R5),-      ;ADD NEGATIVE BYTE COUNT REMAINING
                             7E A5      06CA  1514                  UCB$W_BCNT(R5)          ;...TO PARTIAL TRANSFER COUNT
                   52    00DC C5  D0  06CC  1515          MOVL    UCB$L_DQ_MP(R5),R2      ;FETCH MPR
                                06D1  1516
                                06D1  1517 ;
                                06D1  1518 ; CHECK TO SEE IF THERE IS ANY ERROR OTHER THAN OPI.  IF NO OTHER
                                06D1  1519 ; ERRORS, AND THE DRIVE IS NOT READY, THEN ASSUME IT WAS SIMPLY SPUN DOWN
                                06D1  1520 ;
            51    00807801 8F   D3  06D1  1521          BITL    #RB_CS_M_SSE -          ;SKIP SECTOR ERROR
                                06D8  1522                  !RB_CS_M_DE -           ;... OR DRIVE ERROR
                                06D8  1523                  !RB_CS_M_NXM -          ;... OR NON-EXISTANT MEMORY
                                06D8  1524                  !RB_CS_M_DLT -          ;... OR DATA LATE
                                06D8  1525                  !RB_CS_M_DCK -          ;... OR DATA CHECK
                                06D8  1526                  !RB_CS_M_DRDY,R1        ;... OR DRIVE READY?
                             0D 12  06D8  1527          BNEQ    20$                     ;BRANCH IF SO
                             0B E5  06DA  1528 10$:     BBCC    #UCB$V_VALID,-          ;CLEAR VALID BIT
                      00 64 A5      06DC  1529                  UCB$W_STS(R5),15$       ;...
                   50   01A4 8F   3C  06DF  1530 15$:     MOVZWL  #SS$_MEDOFL,R0          ;SET MEDIUM OFFLINE STATUS
                          FC3A   31  06E4  1531          BRW     FUNCXT                  ;RETURN
                                06E7  1532
                00000000'GF      16  06E7  1533 20$:     JSB     G^ERL$DEVICERR          ;ALLOCATE AND FILL ERROR MESSAGE BUFFER
                  53 51      0D  E0  06ED  1534          BBS     #RB_CS_V_NXM,R1,FATAL   ;BRANCH IF NON-EXISTENT MEMORY
                  4F 51      17  E0  06F1  1535          BBS     #RB_CS_V_SSE,R1,FATAL   ;BRANCH IF SKIP SECTOR ERROR
                  2F 51      0E  E1  06F5  1536          BBC     #RB_CS_V_DE,R1,RETRY    ;BRANCH IF NO DRIVE ERRORS
                                06F9  1537          BDRVTYP RB02,50$                ;BRANCH IF RB02
                                06FF  1538
                                06FF  1539 ;
                                06FF  1540 ;CLASSIFY RB80 ERRORS AS FATAL OR RETRIABLE
                                06FF  1541 ;
                                06FF  1542
                   D7 52        09  E1  06FF  1543          BBC     #RB_MP_V_PLGV,R2,10$    ;BRANCH IF PLUG NOT VALID
                                23  11  0703  1544          BRB     RETRY                   ;
                                0705  1545
                                0705  1546 ;
                                0705  1547 ;CLASSIFY RB02 ERRORS AS FATAL OR RETRIABLE
                                0705  1548 ;
         1D     52       06    00  ED  0705  1549 50$:     CMPZV   #0,#6,R2,-              ;STATE OK? ...COVER CLOSED
                                070A  1550                  #RB_MP_M_HO-            ;...HEADS OUT
                                070A  1551                  !RB_MP_M_BH-            ;...BRUSHES HOME
                                070A  1552                  !RB_MP_C_SLM            ;...READY TO GO
                                CE  12  070A  1553          BNEQ    10$                     ;BRANCH IF NOT
                                070C  1554
                   07 52        09  E1  070C  1555          BBC     #RB_MP_V_VC,R2,60$      ;BRANCH IF VOLUME VALID
                                08  E5  0710  1556          BBCC    #UCB$V_VALID,-          ;CLEAR VALID BIT
```

```
                00 64 A5          0712  1557                        UCB$W_STS(R5),55$           ;...
                      2D    11    0715  1558 55$:     BRB           FATAL                       ;RETURN
                                  0717  1559
              04 52   0D    E1    0717  1560 60$:     BBC           #RB_MP_V_WL,R2,70$          ;BRANCH IF NOT WRITE LOCKED
              25 52   0A    E0    071B  1561           BBS           #RB_MP_V_WGE,R2,FATAL       ;IF WL & WGE THEN WL ERROR
      52   0000CD00   8F    D3    071F  1562 70$:     BITL          #RB_MP_M_WDE-               ;WRITE DATA ERROR
                                  0726  1563                        !RB_MP_M_HCE-               ;...OR HEAD CURRENT ERROR
                                  0726  1564                        !RB_MP_M_SPD-               ;...OR SPINDLE SPEED ERROR
                                  0726  1565                        !RB_MP_M_WGE-               ;...OR WRITE GATE ERROR
                                  0726  1566                        !RB_MP_M_DSE,R2             ;...OR DRIVE SELECT ERROR?
                      1C    12    0726  1567           BNEQ          FATAL                       ;BRANCH IF SO
                                  0728  1568
                                  0728  1569 ;
                                  0728  1570 ; RETRIABLE ERROR EXIT
                                  0728  1571 ;
                                  0728  1572
                      0F    E0    0728  1573 RETRY:    BBS           #IO$V_INHRETRY,-            ;BRANCH IF RETRIES INHIBITED
              16 009A C5          072A  1574                        UCB$W_FUNC(R5),FATAL        ;...
           50 009C D5    98       072E  1575           CVTBL         @UCB$L_DPC(R5),R0           ;GET BRANCH DISPLACEMENT
           50 009C C5    C0       0733  1576           ADDL          UCB$L_DPC(R5),R0            ;COMPUTE JUMP ADDRESS -1
                      50 D6       0738  1577           INCL          R0                          ;COMPUTE JUMP ADDRESS
                      60 17       073A  1578           JMP           (R0)                        ;RETURN TO ERROR ROUTINE
                                  073C  1579
                                  073C  1580 ;
                                  073C  1581 ; SUCCESSFUL OPERATION EXIT
                                  073C  1582 ;
                                  073C  1583
              009C C5    D6       073C  1584 SUCCESS:INCL       UCB$L_DPC(R5)               ;ADJUST TO CORRECT RETURN ADDRESS
              009C D5    17       0740  1585           JMP           @UCB$L_DPC(R5)              ;RETURN TO DRIVER
                                  0744  1586
                                  0744  1587 ;
                                  0744  1588 ; FATAL ERROR EXIT
                                  0744  1589 ;
                                  0744  1590
                FB7B    31        0744  1591 FATAL:    BRW           FATALERR                    ;FATAL ERROR EXIT
                                  0747  1592
```

```
                     1594          .SBTTL  INTERRUPT SERVICE ROUTINE
                     1595
                     1596 ;++
                     1597 ; DQ$INT - RB730 INTERRUPT SERVICE ROUTINE
                     1598 ;
                     1599 ; FUNCTIONAL DESCRIPTION:
                     1600 ;
                     1601 ;       THIS ROUTINE IS ENTERED VIA A JSB INSTRUCTION WHEN AN INTERRUPT
                     1602 ;       OCCURS ON AN RB730 DISK CONTROLLER. IF THE INTERRUPT IS NOT EXPECTED,
                     1603 ;       THE UNSOLICITED INTERRUPT ROUTINE DISMISSES THE INTERRUPT. IF
                     1604 ;       THE INTERRUPT IS EXPECTED, DEVICE REGISTERS ARE SAVED AND THE
                     1605 ;       INTERRUPTING UNIT IS DETERMINED. THE DRIVER IS CALLED AT ITS INTERRUPT
                     1606 ;       RETURN ADDRESS. THE DRIVER FORKS, CAUSING A RETURN TO THIS ROUTINE
                     1607 ;       WHICH CONTINUES TO SCAN THE ATTENTION SUMMARY REGISTER IN CASE
                     1608 ;       ANY MORE DRIVES REQUIRE SERVICE.  AFTER THE LAST DRIVE IS SERVICED
                     1609 ;       THIS ROUTINE RESTORES GENERAL REGISTERS AND DISMISSES THE INTERRUPT.
                     1610 ;
                     1611 ; INPUTS:
                     1612 ;
                     1613 ;       00(SP)  - POINTER TO ADDRESS OF THE IDB
                     1614 ;       04(SP)  - SAVED R0
                     1615 ;       08(SP)  - SAVED R1
                     1616 ;       12(SP)  - SAVED R2
                     1617 ;       16(SP)  - SAVED R3
                     1618 ;       20(SP)  - SAVED R4
                     1619 ;       24(SP)  - SAVED R5
                     1620 ;       28(SP)  - PC AT THE TIME OF THE INTERRUPT
                     1621 ;       32(SP)  - PSL AT THE TIME OF THE INTERRUPT
                     1622 ;
                     1623 ; OUTPUTS:
                     1624 ;
                     1625 ;       DEVICE REGISTERS ARE SAVED, IPL IS LOWERED TO FORK LEVEL, THE
                     1626 ;       INTERRUPT IS DISMISSED, ALL REGISTERS EXCEPT R0-R5 ARE PRESERVED.
                     1627 ;
                     1628 ;--
                     1629
                     1630 DQ_REI:                                          ;INTERRUPT EXIT CODE
     5E    04   C0   1631          ADDL    #4,SP                          ;POP IDB ADDRESS
           3F   BA   1632          POPR    #^M<R0,R1,R2,R3,R4,R5>         ;RESTORE R0-R5
                02   1633          REI                                    ;RETURN FROM INTERRUPT
                     1634
                     1635 DQ_INT:                                 ;INTERRUPT SERVICE ROUTINE
  53   00   BE   D0   1636          MOVL    @(SP),R3                       ;FETCH ADDRESS OF IDB
     54   63   D0   1637          MOVL    IDB$L_CSR(R3),R4               ;GET ADDRESS OF CSR
  55   04   A3   D0   1640          MOVL    IDB$L_OWNER(R3),R5             ;GET OWNER UCB ADDRESS
           0C   13   1643          BEQL    12$                            ;BRANCH IF NOT OWNED
           01   E4   1644          BBSC    #UCB$V_INT,-                   ;BRANCH IF INTERRUPT EXPECTED
     47 64   A5      1645                  UCB$W_STS(R5),40$              ;...
                     1646 ;
                     1647 ;SCAN ATTENTION BITS TO DETERMINE INTERRUPTING DRIVE
                     1648 ;
  53   00   BE   D0   1649 10$:     MOVL    @(SP),R3                       ;FETCH ADDRESS OF IDB
     54   63   D0   1650          MOVL    IDB$L_CSR(R3),R4               ;GET ADDRESS OF CSR
     51   64   D0   1651 12$:     MOVL    RB_CS(R4),R1                   ;GET CSR
           10   EA   1652          FFS     #RB_CS_V_ATN,-                 ;FIND REQUESTING DRIVE
51   51   04      1653                  #RB_CS_S_ATN,R1,R1             ;...
           D7   13   1654          BEQL    DQ_REI                         ;BRANCH IF NO MORE DRIVES TO SERVICE
```

```
                50    0F    10    78  0770  1651          ASHL    #RB_CS_V_ATN,#^X0F,R0   ;PREPARE MASK OF ATTENTION BITS
                  00  50    51    E5  0774  1652          BBCC    R1,R0,T5$               ;CLEAR THIS UNIT'S BIT IN THE MASK
                      64    50    CA  0778  1653 15$:     BICL    R0,RB_CS(R4)            ;CLEAR THIS UNIT'S BIT IN THE CSR
                      51    10    C2  077B  1654          SUBL    #RB_CS_V_ATN,R1         ;COMPUTE UNIT NUMBER
                52    51    08    78  077E  1655          ASHL    #RB_CS_V_DS,R1,R2       ;MOVE UNIT INTO DRIVE SELECT BITS
                            52    C9  0782  1656          BISL3   R2,-                    ;SELECT THE UNIT
                                      0784  1657                  #RB_CS_M_IE-            ;...WITH INTERRUPT ENABLE
                                      0784  1658                  !RB_CS_M_CRDY,-         ;...AND CONTROLLER READY
            64   000000C0 8F          0784  1659                  RB_CS(R4)               ;...DEVICE CSR
                55    18 A341   D0    078A  1660          MOVL    IDB$L_UCBLST(R3)[R1],R5 ;GET ADDRESS OF UCB
                            10    13  078F  1661          BEQL    25$                     ;BRANCH IF UCB WAS NOT FOUND
                            01    E4  0791  1662          BBSC    #UCB$V_INT,-            ;BRANCH IF INTERRUPT WAS EXPECTED
                      10 64 A5          0793  1663                  UCB$W_STS(R5),40$       ;...
                00CC C5   64    D0    0796  1664          MOVL    RB_CS(R4),UCB$L_DQ_CS(R5);SAVE CSR
                            00    E4  079B  1665          BBSC    #UCB$V_DQ_SIP,-         ;BRANCH IF SEEK IN PROGRESS
                  C5 00C9 C5          079D  1666                  UCB$B_DQ_FLAGS(R5),12$  ;...AND CONTINUE SCANNING
                                      07A1  1667 25$:
                         00C8   30    07A1  1668          BSBW    DQ_UNEXINT              ;HANDLE UNEXPECED INTERRUPT
                            C0    11  07A4  1669          BRB     12$                     ;CONTINUE SCANNING
                                      07A6  1670
                                      07A6  1671 ;
                                      07A6  1672 ;HERE WHEN UNIT DETERMINED, INTERRUPT EXPECTED, DRIVE SELECTED
                                      07A6  1673 ;AND STATUS AVAILABLE
                                      07A6  1674 ;
                0093 C5     0E    91    07A6  1675 40$:     CMPB    #CDF_READHEAD,UCB$B_CEX(R5)     ;READ HEADER FUNCTION?
                            12    12    07AB  1676          BNEQ    50$                     ;IF NEQ - NO
             00EC C5    10 A4    F7    07AD  1677          CVTLW   RB_MP(R4),UCB$W_DQ_HDR1(R5)     ;SAVE SECTOR HEADER INFORMATION
             00EE C5    10 A4    F7    07B3  1678          CVTLW   RB_MP(R4),UCB$W_DQ_HDR2(R5)     ;...(THIS MUST BE DONE EVEN
             00F0 C5    10 A4    F7    07B9  1679          CVTLW   RB_MP(R4),UCB$W_DQ_HDR3(R5)     ;...FOR INTERNAL READ HEADERS)
                                      07BF  1680
            64   00008000 8F    D3    07BF  1681 50$:     BITL    #RB_CS_M_CE,RB_CS(R4)           ;COMPOSITE ERROR?
                            14    12  07C6  1682          BNEQ    80$                     ;BRANCH IF SO
                            01    E0  07C8  1683          BBS     #UCB$V_DIAGBUF,-        ;BRANCH IF DIAGNOSTIC BUFFER
                      0F 68 A5          07CA  1684                  UCB$W_DEVSTS(R5),80$    ;...IS PRESENT
                00CC C5     64    D0    07CD  1685          MOVL    RB_CS(R4),UCB$L_DQ_CS(R5)       ;SAVE CSR ONLY
                                      07D2  1686
                                      07D2  1687 ;
                                      07D2  1688 ;RETURN TO FUNCTION EXECUTION
                                      07D2  1689 ;
                  53    10 A5     7D    07D2  1690 60$:     MOVQ    UCB$L_FR3(R5),R3        ;RESTORE DRIVER CONTEXT
                            0C B5   16    07D6  1691          JSB     @UCB$L_FPC(R5)          ;CALL DRIVER AT INTERRUPT RETURN ADDRESS
                             FF83   31    07D9  1692          BRW     10$                     ;CHECK FOR MORE DRIVES TO SERVICE
                                      07DC  1693
                                      07DC  1694 ;
                                      07DC  1695 ;DEVICE ERROR OR DIAGNOSTIC BUFFER -- SAVE THE DEVICE REGISTERS
                                      07DC  1696 ;AND RESET THE DRIVE
                                      07DC  1697 ;
                            02    10    07DC  1698 80$:     BSBB    DQ_REGSAVE              ;SAVE DEVICE REGISTERS
                            F2    11    07DE  1699          BRB     60$                     ;CONTINUE
                                      07E0  1700
```

```
                     07E0  1702          .SBTTL  REGISTER SAVE ROUTINE
                     07E0  1703 ;++
                     07E0  1704 ;
                     07E0  1705 ; DQ_REGSAVE - REGISTER SAVE ROUTINE
                     07E0  1706 ;
                     07E0  1707 ; FUNCTIONAL DESCRIPTION:
                     07E0  1708 ;
                     07E0  1709 ; THIS ROUTINE IS CALLED TO SAVE THE DEVICE REGISTERS AND UBA RESOURCE
                     07E0  1710 ; REGISTERS IN THE UCB.  STATUS IS OBTAINED FROM THE DRIVE AND IF AN
                     07E0  1711 ; ERROR HAS OCCURED THEN THE DRIVE IS RESET (COULD BE HERE BECAUSE
                     07E0  1712 ; DIAGNOSTIC BUFFER PRESENT.
                     07E0  1713 ;
                     07E0  1714 ; INPUTS:
                     07E0  1715 ;
                     07E0  1716 ;       R4      - ADDRESS OF DEVICE CONTROL STATUS REGISTER (CSR)
                     07E0  1717 ;       R5      - ADDRESS OF UNIT CONTROL BLOCK (UCB)
                     07E0  1718 ;
                     07E0  1719 ; OUTPUTS:
                     07E0  1720 ;
                     07E0  1721 ;       R0-R3   - DESTROYED
                     07E0  1722 ;       THE DEVICE REGISTERS ARE SAVED IN THE UCB.
                     07E0  1723 ;
                     07E0  1724 ;--
                     07E0  1725
                     07E0  1726 DQ_REGSAVE:                          ;REGISTER SAVE ROUTINE
     52   64   9E   07E0  1727          MOVAB   RB_CS(R4),R2            ;GET ADDRESS OF CONTROL STATUS REGISTER
53   00CC C5   9E   07E3  1728          MOVAB   UCB$L_DQ_CS(R5),R3      ;GET ADDRESS OF REGISTER SAVE AREA
     83   82   D0   07E8  1729          MOVL    (R2)+,(R3)+             ;SAVE CONTROL STATUS REGISTER
     83   82   D0   07EB  1730          MOVL    (R2)+,(R3)+             ;SAVE BUFFER ADDRESS REGISTER
     83   82   D0   07EE  1731          MOVL    (R2)+,(R3)+             ;SAVE BYTE COUNT REGISTER
     83   82   D0   07F1  1732          MOVL    (R2)+,(R3)+             ;SAVE DISK ADDRESS REGISTER
     83   82   D0   07F4  1733          MOVL    (R2)+,(R3)+             ;SAVE MPR REGISTER
00C4 C5   82   F7   07F7  1734          CVTLW   (R2)+,UCB$W_EC1(R5)     ;SAVE ECC POSITION REGISTER
00C6 C5   82   F7   07FC  1735          CVTLW   (R2)+,UCB$W_EC2(R5)     ;SAVE ECC PATTERN REGISTER
                     0801  1736 ;
                     0801  1737 ; GET STATUS
                     0801  1738 ;
                     0801  1739          GETUNIT                         ;GET UNIT NUMBER IN R2
          01   ED   0809  1740          CMPZV   #RB_CS_V_FCODE,-        ;WAS ORIGIANL FUNCTION A GET STATUS?
          03        080B  1741                  #RB_CS_S_FCODE,-        ;...
     00CC C5        080C  1742                  UCB$L_DQ_CS(R5),-       ;...
          04        080F  1743                  #F_GETSTATUS            ;...
          12   13   0810  1744          BEQL    20$                     ;BRANCH IF SO (USE ORIGINAL STATUS)
00DC C5   FF  8F   9A   0812  1745          MOVZBL  #-1,UCB$L_DQ_MP(R5)     ;SET TO -1 IF GET STATUS FAILS
        00A6   30   0818  1746          BSBW    DQ_GETSTS               ;GET THE STATUS
      06  50   E9   081B  1747          BLBC    R0,20$                  ;BRANCH IF GET STATUS FAILED
00DC C5   10  A4   D0   081E  1748          MOVL    RB_MP(R4),UCB$L_DQ_MP(R5);SAVE MPR (STATUS WORD)
          0F   E1   0824  1749 20$:       BBC     #RB_CS_V_CE,-           ;BRANCH IF NO ERRORS (DON'T CLEAR IF
   03 00CC C5        0826  1750                  UCB$L_DQ_CS(R5),30$     ;...ONLY HERE FOR DIAGNOSTIC BUFFER)
        008E   30   082A  1751          BSBW    DQ_RESET                ;CLEAR DRIVE ERRORS IF ANY
                     082D  1752
                     082D  1753
                     082D  1754 ;
                     082D  1755 ; SAVE UBA REGISTERS
                     082D  1756 ;
                     082D  1757
                     082D  1758 30$:     ASSUME  UCB$L_DQ_FMPR EQ UCB$L_DQ_MP+4 ;ASSUME REGISTER AREA CONTIG
```

```
     63  00000000 FFFFFFFF 8F    7D  082D  1759          MOVQ    #-1,(R3)                ;SET DEFAULT VALUE -1
                        0A    91  0838  1760          CMPB    #CDF_WRITECHECK,-       ;DRIVE RELATED FUNCTION?
                      0093 C5       083A  1761                  UCB$B_CEX(R5)           ;...
                        2C    1A  083D  1762          BGTRU   70$                     ;BRANCH IF SO
                                  083F  1763
         50   00D0 C5    F7 8F    78  083F  1764          ASHL    #-9,UCB$L_DQ_BA(R5),R0  ;COMPUTE MAP REGISTER NUMBER
              50    01EF 8F       B1  0846  1765          CMPW    #495,R0                 ;LEGAL MAP REGISTER NUMBER?
                        1E    1F  084B  1766          BLSSU   70$                     ;BRANCH IF NOT
                                  084D  1767
                51    24 A5     D0  084D  1768          MOVL    UCB$L_CRB(R5),R1        ;FETCH CRB
                52    38 B1     D0  0851  1769          MOVL    @CRB$L_INTD+VEC$L_ADP(R1),R2 ;FETCH ADDRESS OF ADAPTOR CSR
           83    0800 C240      D0  0855  1770          MOVL    UBI$L_MAP(R2)[R0],(R3)+ ;SAVE FINAL MAP REGISTER
                        50    D7  085B  1771          DECL    R0                      ;CALCULATE PREVIOUS MAP REGISTER NUMBER
                        00    EC  085D  1772          CMPV    #VEC$V_MAPREG,-         ;COMPARE STARTING MAP REGISTER NUMBER
                        0F        085F  1773                  #VEC$S_MAPREG,-         ;...
                      34 A1       0860  1774                  CRB$L_INTD+VEC$W_MAPREG(R1),- ;...FROM CRB
                        50        0862  1775                  R0                      ;...TO ENDING MAP REGISTER MINUS 1
                        06    14  0863  1776          BGTR    70$                     ;BRANCH IF WE'RE STILL ON FIRST MAP
           63    0800 C240      D0  0865  1777          MOVL    UBI$L_MAP(R2)[R0],(R3)  ;SAVE PREVIOUS MAP REGISTER
                        05    05  086B  1778 70$:      RSB                             ;RETURN
                                  086C  1779
```

```
                     086C  1781          .SBTTL  UNEXEPECTED INTERRUPT HANDLER
                     086C  1782 ;++
                     086C  1783 ;
                     086C  1784 ; FUNCTIONAL DESCRIPTION:
                     086C  1785 ;
                     086C  1786 ; UNEXPECTED INTERRUPTS ARE FIELDED.  STATUS IS OBTAINED FROM
                     086C  1787 ; THE DRIVE, VOLUME VALID IS CLEARED IF APPROPRIATE AND THE
                     086C  1788 ; DRIVE IS RESET
                     086C  1789
                     086C  1790 ; INPUTS:
                     086C  1791 ;
                     086C  1792 ;       R2      - UNIT NUMBER OF DRIVE
                     086C  1793 ;       R4      - RB730 CSR ADDRESS
                     086C  1794 ;       R5      - UCB ADDRESS
                     086C  1795 ;
                     086C  1796 ; OUTPUTS:
                     086C  1797 ;
                     086C  1798 ;       R0      - DESTROYED
                     086C  1799 ;       UCB$V_VALID IN UCB$W_STS
                     086C  1800 ;
                     086C  1801 ;--
                     086C  1802
                     086C  1803 DQ_UNEXINT:                         ;UNEXPECTED INTERRUPT
              55  D5   086C  1804          TSTL    R5                      ;IS THERE AN UCB?
              30  13   086E  1805          BEQL    50$                     ;BRANCH IF NOT
                     0870  1806
              4F  10   0870  1807          BSBB    DQ_GETSTS               ;GET DRIVE STATUS
            23 50  E9   0872  1808          BLBC    R0,20$                  ;BRANCH IF TIMEOUT OR OPI
         64 A5  10   A8 0875  1809          BISW    #UCB$M_ONLINE,UCB$W_STS(R5)     ;SET UCB STATUS VOLUME ONLINE
                     0879  1810
      64 04000000 8F   D3 0879  1811          BITL    #RB_CS_M_TYP,RB_CS(R4)  ;IS THIS AN R80
                  0C   12 0880  1812          BNEQ    10$                     ;BRANCH IF SO
  10 A4 00000200 8F   D3 0882  1813          BITL    #RB_MP_M_VC,RB_MP(R4)   ;VOLUME CHECK?
                  11   13 088A  1814          BEQL    30$                     ;BRANCH IF NOT
                  0A   11 088C  1815          BRB     20$                     ;SET VOLUME INVALID
                     088E  1816
  10 A4 00000200 8F   D3 088E  1817 10$:     BITL    #RB_MP_M_PLGV,RB_MP(R4) ;PLUG VALID?
                  05   12 0896  1818          BNEQ    30$                     ;BRANCH IF SO
                     0898  1819
                  0B   E5 0898  1820 20$:     BBCC    #UCB$V_VALID,-          ;CLEAR VALID BIT
            00 64 A5      089A  1821                  UCB$W_STS(R5),30$       ;...
                     089D  1822
                00DA   30 089D  1823 30$:     BSBW    DQ_CLASSIFY             ;CLASSIFY DRIVE AND INIT UCB
                     08A0  1824
                  19   10 08A0  1825 50$:     BSBB    DQ_RESET                ;RESET THE DRIVE
                  05   05 08A2  1826          RSB                             ;RETURN TO CALLER
                     08A3  1827
```

```
                     08A3  1829          .SBTTL  GET STATUS, RESET, READ HEADER
                     08A3  1830 ;++
                     08A3  1831 ;
                     08A3  1832 ; DQ_READHDR - READ HEADER (EITHER DRIVE)
                     08A3  1833 ; DQ_RESET   - GET STATUS AND RESET ROUTINE
                     08A3  1834 ; DQ_GETSTS  - GET STATUS ROUTINE
                     08A3  1835 ;
                     08A3  1836 ; FUNCTIONAL DESCRIPTION:
                     08A3  1837 ;
                     08A3  1838 ; THIS ROUTINE HANDLES NON-INTERRUPT DRIVEN DEVICE OPERATIONS INCLUDING:
                     08A3  1839 ;
                     08A3  1840 ;       RESET DRIVE
                     08A3  1841 ;       GET STATUS
                     08A3  1842 ;       READ HEADER
                     08A3  1843 ;
                     08A3  1844 ; AFTER EXECUTING THE FUNCTION A WAIT FOR CONTROLLER READY IS DONE.
                     08A3  1845 ; THE WAIT WILL TIMEOUT IF CONTROLLER READY DOES NOT APPEAR WITHIN
                     08A3  1846 ; 2 SECONDS
                     08A3  1847 ;
                     08A3  1848 ; THIS ROUTINE SHOULD ONLY BE CALLED AT DEVICE IPL OR ABOVE
                     08A3  1849 ;
                     08A3  1850 ; INPUTS:
                     08A3  1851 ;
                     08A3  1852 ;       R2      - UNIT NUMBER IN DRIVE SELECT BITS
                     08A3  1853 ;       R4      - ADDRESS OF DEVICE CONTROL STATUS REGISTER (CSR)
                     08A3  1854 ;
                     08A3  1855 ; OUTPUTS:
                     08A3  1856 ;
                     08A3  1857 ;       RB_MP(R4)       - DRIVE STATUS IF DQ_GETSTS OR DQ_RESET
                     08A3  1858 ;       R0              - LOW BIT CLEAR IF A TIMEOUT OR OPERATION INCOMPLETE
                     08A3  1859 ;
                     08A3  1860 ;
                     08A3  1861 ;--
                     08A3  1862
                     08A3  1863 ;
                     08A3  1864 ;TO READ A HEADER, THE COMMAND IS LOADED AND THE WAIT ROUTINE
                     08A3  1865 ;IS JUMPED TO.
                     08A3  1866 ;
                     08A3  1867 ;THE RB730 HOST (VAX730) MICROCODE MAINTAINS AN INTERNAL RECORD
                     08A3  1868 ;OF THE CURRENT DISK CYLINDER FOR RB02'S.  THIS REGISTER IS USED TO
                     08A3  1869 ;COMPUTE THE RELATIVE CYLINDER ADDRESSES REQUIRED BY THE DRIVE.  THE
                     08A3  1870 ;CONTENTS OF THIS REGISTER MAY DISAGREE WITH THE ACTAUL DISK POSITION.
                     08A3  1871 ;
                     08A3  1872 ;THE REGISTER IS RECALIBRATED BY DOING A READ HEADER.  THE MICROCODE
                     08A3  1873 ;RELOADS THE REGISTER WITH THE CURRENT CYLINDER ADDRESS AS SPECIFIED
                     08A3  1874 ;IN THE HEADER WORD, WHEN THE MACRO CODE READS THE MPR.
                     08A3  1875 ;
                     08A3  1876
                     08A3  1877 DQ_READHDR:                                  ;DRIVE READ HEADER ENTRY
                  3C 10   08A3  1878          BSBB    DQ_WAIT                     ;MAKE SURE CONTROLLER FREE
               01 50 E8   08A5  1879          BLBS    R0,10$                      ;BRANCH IF SO
                     05   08A8  1880          RSB                                 ;RETURN WITH R0 LBC
                  52 C9   08A9  1881 10$:     BISL3   R2,-                        ;MERGE UNIT NUMBER
                          08AB  1882                  #F_READHEAD-                ;...WITH FUNCTION
                          08AB  1883                  !RB_CS_M_IE,-               ;...AND INTERRUPT ENABLE
64 00000048 8F            08AB  1884                  RB_CS(R4)                   ;INTO CSR CLEARING CRDY
              002D 30     08B1  1885          BSBW    DQ_WAIT                     ;WAIT FOR COMPLETION
```

```
                10 A4   10 A4   D1   08B4  1886          CMPL    RB_MP(R4),RB_MP(R4)             ;READ HEADER WORDS
                        18      11   08B9  1887          BRB     CHECKOPI                        ;CHECK FOR COMPLETION
                                     08BB  1888 ;
                                     08BB  1889 ;TO RESET THE DRIVE, A GET STATUS SUBCOMMAND IS LOADED INTO THE
                                     08BB  1890 ;MULTIPURPOSE REGISTER WITH THE RESET BIT SET.
                                     08BB  1891 ;
                                     08BB  1892
                                     08BB  1893 DQ_RESET:                                        ;DRIVE RESET ENTRY
                    10 A4   0B   D0  08BB  1894          MOVL    #RB_MP_M_STS!-                  ;PUT GET STATUS IN MPR
                                     08BF  1895                  RB_MP_M_RST!-                   ;...AND RESET THE DRIVE
                                     08BF  1896                  RB_MP_M_MRK,RB_MP(R4)           ;...MARK SUBCOMMAND PRESENT
                        04      11   08BF  1897          BRB     EXGETSTS                        ;CONTINUE IN COMMON
                                     08C1  1898
                                     08C1  1899 ;
                                     08C1  1900 ;TO GET STATUS WITHOUT RESET, A GET STATUS SUBCOMMAND IS LOADED INTO
                                     08C1  1901 ;THE MULTIPURPOSE REGISTER.  DRIVE STATUS IS NOT RESET
                                     08C1  1902 ;
                                     08C1  1903
                                     08C1  1904 DQ_GETSTS:                                       ;GET STATUS ENTRY
                    10 A4   03   D0  08C1  1905          MOVL    #RB_MP_M_STS!-                  ;PUT GET STATUS IN MPR
                                     08C5  1906                  RB_MP_M_MRK,RB_MP(R4)           ;...MARK SUBCOMMAND PRESENT
                                     08C5  1907 ;
                                     08C5  1908 ;NOW EXECUTE THE ACTUAL COMMAND BY MERGING THE UNIT NUMBER WITH
                                     08C5  1909 ;THE GET STATUS COMMAND AND LOADING THE CSR.  INTERUPTS ARE NOT ENABLED
                                     08C5  1910 ;
                                     08C5  1911
                                     08C5  1912 EXGETSTS:                                        ;COMMAND EXECUTION
                        1A      10   08C5  1913          BSBB    DQ_WAIT                         ;MAKE SURE CONTROLLER FREE
                    01 50       E8   08C7  1914          BLBS    R0,10$                          ;BRANCH IF SO
                                05   08CA  1915          RSB                                     ;RETURN WITH R0 LBC
                            52  C9   08CB  1916 10$:     BISL3   R2,-                            ;MERGE UNIT NUMBER
                                     08CD  1917                  #F_GETSTATUS-                   ;...WITH FUNCTION
                                     08CD  1918                  !RB_CS_M_IE,-                   ;...AND INTERRUPT ENABLE
                64  00000044   8F       08CD  1919                  RB_CS(R4)                       ;INTO CSR CLEARING CRDY
                                     08D3  1920
                                     08D3  1921 CHECKOPI:                                        ;CHECK FOR OPERATION INCOMPLETE
                        0C      10   08D3  1922          BSBB    DQ_WAIT                         ;WAIT FOR READY
                64  00000400   8F   D3   08D5  1923          BITL    #RB_CS_M_OPI,RB_CS(R4)          ;OPERATION COMPLETE?
                        02      13   08DC  1924          BEQL    10$                             ;BRANCH IF SO
                        50      D4   08DE  1925          CLRL    R0                              ;SET FAILURE
                                05   08E0  1926 10$:     RSB                                     ;R0 LBC IF TIMEOUT
                                     08E1  1927
```

```
                     08E1  1929          .SBTTL  WAIT FOR CONTROLLER READY
                     08E1  1930 ;++
                     08E1  1931 ;
                     08E1  1932 ; DQ_WAIT    - WAIT FOR CONTROLLER READY ROUTINE
                     08E1  1933 ;
                     08E1  1934 ; FUNCTIONAL DESCRIPTION:
                     08E1  1935 ;
                     08E1  1936 ; THIS ROUTINE WAIT FOR CONTROLLER READY -- THE WAIT WILL TIMEOUT
                     08E1  1937 ; IF CONTROLLER READY DOES NOT APPEAR WITHIN 2 SECONDS
                     08E1  1938 ;
                     08E1  1939 ; THIS ROUTINE SHOULD ONLY BE CALLED AT DEVICE IPL OR ABOVE
                     08E1  1940 ;
                     08E1  1941 ; INPUTS:
                     08E1  1942 ;
                     08E1  1943 ;       R2      - UNIT NUMBER IN DRIVE SELECT BITS
                     08E1  1944 ;       R4      - ADDRESS OF DEVICE CONTROL STATUS REGISTER (CSR)
                     08E1  1945 ;
                     08E1  1946 ; OUTPUTS:
                     08E1  1947 ;
                     08E1  1948 ;       R0              - LOW BIT CLEAR IF A TIMEOUT
                     08E1  1949 ;
                     08E1  1950 ;
                     08E1  1951 ;--
                     08E1  1952
                     08E1  1953 ;
                     08E1  1954 ; WAIT FOR CONTROLLER READY.  IF NOT PRESENT WITHIN APPROXIMATELY
                     08E1  1955 ; 2 SECONDS, THEN R0 WILL HAVE LOW BIT CLEAR
                     08E1  1956 ;
                     08E1  1957 DQ_WAIT:                                    ;WAIT FOR CONTROLLER READY
                     08E1  1958          TIMEWAIT #200000,#RB_CS_M_CRDY,-   ;WAIT FOR CONTROLLER READY
                     08E1  1959                  RB_CS(R4),L                ;... 200000*10 MICS
                  05 090C  1960          RSB                                ;RETURN TO CALLER
                     090D  1961
```

```
                                090D  1963          .SBTTL  UNIT INITIALIZATION ROUTINE
                                090D  1964
                                090D  1965 ;++
                                090D  1966 ;
                                090D  1967 ; DQ_UNIT_INIT - UNIT INITIALIZATION ROUTINE
                                090D  1968 ;
                                090D  1969 ; FUNCTIONAL DESCRIPTION:
                                090D  1970 ;
                                090D  1971 ;       THIS ROUTINE READIES THE RB02/RB80 UNITS FOR I/O OPERATIONS.
                                090D  1972 ;
                                090D  1973 ;       THE OPERATING SYSTEM CALLS THIS ROUTINE:
                                090D  1974 ;               - AT SYSTEM STARTUP
                                090D  1975 ;               - DURING DRIVER LOADING
                                090D  1976 ;               - DURING RECOVERY FROM POWER FAILURE
                                090D  1977 ;
                                090D  1978 ; INPUTS:
                                090D  1979 ;
                                090D  1980 ;       R4      - CSR ADDRESS (CONTROLLER STATUS REGISTER)
                                090D  1981 ;       R5      - UCB ADDRESS (UNIT CONTROL BLOCK)
                                090D  1982 ;
                                090D  1983 ; OUTPUTS:
                                090D  1984 ;
                                090D  1985 ;       THE DRIVE IS RESET, UCB FIELDS ARE INITIALIZED, AND THE
                                090D  1986 ;       ROUTINE WAITS FOR ONLINE UNITS TO SPIN UP.  ALL REGISTERS
                                090D  1987 ;       EXCEPT R0-R3 ARE PRESERVED.  RB80'S ARE DIFFERENTIATED FROM
                                090D  1988 ;       RB02 UNITS FOR UCB INITIALIZATION PURPOSES
                                090D  1989 ;
                                090D  1990 ;       A PERMANENT BUFFERED DATAPATH AND A PERMANENT SET OF MAP
                                090D  1991 ;       REGISTERS ARE ALLOCATED ON THE FIRST CALL TO THIS ROUTINE.
                                090D  1992 ;       ON SUCESSIVE ENTRYS, THE CALLS TO ALLOCATE RESOURCES ARE
                                090D  1993 ;       IGNORED BY THE SYSTEM.
                                090D  1994 ;--
                                090D  1995
                                090D  1996
                                090D  1997 DQ_UNIT_INIT:                                ;RB02/RB80 UNIT INITIALIZATION
                                090D  1998
                                090D  1999 ;
                                090D  2000 ; GET CURRENT DRIVE STATUS AND RESET DRIVE
                                090D  2001 ;
 53   64 A5    3C               090D  2002          MOVZWL  UCB$W_STS(R5),R3            ;SAVE CURRENT UNIT STATUS
    0810 BF    AA               0911  2003          BICW    #UCB$M_ONLINE!UCB$M_VALID,- ;ASSUME OFFLINE/INVALID
      64 A5                     0915  2004                  UCB$W_STS(R5)               ;...
                                0917  2005          GETUNIT                             ;LOAD UNIT NUMBER IN R2
         C0    10               091F  2006          BSBB    DQ_WAIT                     ;WAIT FOR CONTROLLER
       26 50   E9               0921  2007          BLBC    R0,50$                      ;BRANCH IF CONTROLLER BUSY
        FF94   30               0924  2008          BSBW    DQ_RESET                    ;GET STATUS AND RESET DRIVE
       20 50   E9               0927  2009          BLBC    R0,50$                      ;BRANCH IF TIMEOUT OR OPI
                                092A  2010
                                092A  2011 ;
                                092A  2012 ; WAIT FOR ONLINE UNITS TO SPIN UP
                                092A  2013 ;
                                092A  2014
 16 53    08    E1               092A  2015          BBC     #UCB$V_VALID,R3,40$         ;BYPASS SPINUP WAIT IF NOT
                                092E  2016                                              ;...VALID BEFORE POWER FAIL
    64    01    D3               092E  2017 10$:     BITL    #RB_CS_M_DRDY,RB_CS(R4)     ;IS DRIVE READY?
          08    12               0931  2018          BNEQ    30$                         ;BRANCH IF READY
00000000'GF     16               0933  2019          JSB     G^EXE$PWRTIMCHK             ;IS MAX TIME EXCEEDED?
```

```
                F2 50    E8  0939  2020          BLBS    R0,10$                          ;IF LBS - NO, STILL MORE TIME NEEDED
                   06    11  093C  2021          BRB     40$                             ;POWER UP TIME EXCEEDED
                         093E  2022
          64 A5  0800 8F   A8  093E  2023 30$:    BISW    #UCB$M_VALID,UCB$W_STS(R5)      ;SET UCB STATUS VOLUME VALID
              64 A5    10   A8  0944  2024 40$:    BISW    #UCB$M_ONLINE,UCB$W_STS(R5)     ;SET UCB STATUS VOLUME ONLINE
                   30    10  0948  2025          BSBB    DQ_CLASSIFY                     ;CLASSIFY DRIVE
                         094A  2026
                         094A  2027
                         094A  2028 ;
                         094A  2029 ; ALLOCATE A PERMANENT BUFFERED DATAPATH.  (ON A VAX730, ALL DATAPATHS
                         094A  2030 ; ARE DIRECT.  IT IS ALLOCATED HERE ONLY OUT OF CONVENTION.)
                         094A  2031 ;
                         094A  2032 50$:    REQDPRNW                                ;REQUEST A PATH -- NO WAIT
                04 50    E8  0950  2033          BLBS    R0,55$                          ;BRANCH IF SUCCESSFUL
                         0953  2034          BUG_CHECK UBMAPEXCED,FATAL              ;SERIOUS PROBLEM
             51  24 A5    D0  0957  2035 55$:    MOVL    UCB$L_CRB(R5),R1                ;FETCH CRB ADDRESS
                   07    E2  095B  2036          BBSS    #VEC$V_PATHLOCK,-               ;LOCK THE DATA PATH
               00 37 A1        095D  2037                  CRB$L_INTD+VEC$B_DATAPATH(R1),65$ ;...IN THE CRB
                         0960  2038
                         0960  2039 ;
                         0960  2040 ; ACCOCATE ENOUGH PERMANENT MAP REGISTERS TO HOLD THE LARGEST POSSIBLE
                         0960  2041 ; DATA TRANSFER.  SINCE NEITHER THE RB02 OR RB80 SUPPORT SPIRALLING THE
                         0960  2042 ; LARGEST TRANSFER IS A SINGLE TRACK
                         0960  2043 ;
                         0960  2044 ;       32 (RB80 BLOCKS PER TRACK) +
                         0960  2045 ;        1 (IN CASE TRANSFER CROSSES PAGE BOUND) +
                         0960  2046 ;        1 (FOR INVALID SENTINAL PAGE)
                         0960  2047 ;       -- =
                         0960  2048 ;       34 PERMANENTLY ALLOCATED MAP REGISTERS
                         0960  2049 ;
                53   22  D0  0960  2050 65$:    MOVL    #34,R3                          ;34 MAP REGISTERS NEEDED
          00000000'GF   16  0963  2051          JSB     G^IOC$ALOUBAMAPN                ;REQUEST THEM
                         0969  2052
                04 50    E8  0969  2053          BLBS    R0,67$                          ;BRANCH IF SUCCESSFUL
                         096C  2054          BUG_CHECK UBMAPEXCED,FATAL              ;SERIOUS PROBLEM
                         0970  2055
             51  24 A5    D0  0970  2056 67$:    MOVL    UCB$L_CRB(R5),R1                ;FETCH CRB ADDRESS
                   0F    E2  0974  2057          BBSS    #VEC$V_MAPLOCK,-                ;LOCK THE MAPS
               00 34 A1        0976  2058                  CRB$L_INTD+VEC$W_MAPREG(R1),75$ ;...IN THE CRB
                   05    0979  2059 75$:    RSB                                     ;RETURN
                         097A  2060
```

```
                                        .SBTTL  DRIVE CLASSIFICATION ROUTINE
                                ;++
                                ;
                                ; DQ_CLASSIFY - DRIVE CLASSIFICATION ROUTINE
                                ;
                                ; FUNCTIONAL DESCRIPTION:
                                ;
                                ; THIS ROUTINE IS CALLED TO CLASSIFY THE DRIVE TYPE AND INITIALIZE
                                ; THE UCB FEILDS.  IT IS CALLED AT DRIVE INIT TIME, AND FOLLOWING AN
                                ; UNEXPECT INTERRUPT.
                                ;
                                ; INPUTS:
                                ;
                                ;       R4      - ADDRESS OF DEVICE CONTROL STATUS REGISTER (CSR)
                                ;       R5      - ADDRESS OF UNIT CONTROL BLOCK (UCB)
                                ;
                                ; OUTPUTS:
                                ;
                                ;       R0-R2   - DESTROYED
                                ;       THE UCB FEILDS ARE INITIALIZED
                                ;
                                ;--
                                
                                DQ_CLASSIFY:                            ;DRIVE CLASSIFICATION ROUTINE
                                ;
                                ; ASSUME THAT SECTORS, TRACKS, AND CYLINDERS FEILDS ARE CONTAINED IN
                                ; UCB$L_DEVDEPEND
                                ;
                                ASSUME  UCB$B_SECTORS   EQ      UCB$L_DEVDEPEND
                                ASSUME  UCB$B_TRACKS    EQ      UCB$L_DEVDEPEND+1
                                ASSUME  UCB$W_CYLINDERS EQ      UCB$L_DEVDEPEND+2
                                
                                ;
                                ; ASSUME ITS AN RB02 AND INITIALIZE ACCORDINGLY
                                ;
          41 A5    12    90                 MOVB    #DT$_RB02,UCB$B_DEVTYPE(R5)     ;SET RB02 DEVICE TYPE AND
        02000228 8F    D0                 MOVL    #<40*<2@8>+<512@16>>,-          ;LOAD SECTORS+TRACKS+CYLINDERS
             44 A5                               UCB$L_DEVDEPEND(R5)             ;...INTO UCB
    0080 C5    5000 8F    3C                 MOVZWL  #<20*2*512>,UCB$L_MAXBLOCK(R5)  ;(512 BYTE) BLOCKS PER SPINDLE
008C C5    24642002 8F    D0                 MOVL    #^X24642002,UCB$L_MEDIA_ID(R5)  ;SET MEDIA IDENT "DQ RB02"
          68 A5    04    A8                 BISW2   #UCB$M_NOCNVRT,UCB$W_DEVSTS(R5) ;DISABLE LOG TO PHYS CONV.
                                            GETUNIT                                 ;PUT UNIT NUMBER IN R2
                   FEFE    30                 BSBW    DQ_READHDR                      ;READ HEADER TO SYNCRONIZE UCODE
                                
     64    04000000 8F    D3                 BITL    #RB_CS_M_TYP,RB_CS(R4)          ;TEST DRIVE TYPE
                   22    13                 BEQL    30$                             ;BRANCH IF AN RB02
                                
          41 A5    13    90                 MOVB    #DT$_RB80,UCB$B_DEVTYPE(R5)     ;SET RB80 DEVICE TYPE AND
        022F0E1F 8F    D0                 MOVL    #<31*<14@8>+<559@16>>,-          ;LOAD SECTORS+TRACKS+CYLINDERS
             44 A5                               UCB$L_DEVDEPEND(R5)             ;...INTO UCB
0080 C5    000383AE 8F    D0                 MOVL    #<31*14*559>,UCB$L_MAXBLOCK(R5) ;(512 BYTE) BLOCKS PER SPINDLE
008C C5    24642050 8F    D0                 MOVL    #^X24642050,UCB$L_MEDIA_ID(R5)  ;SET MEDIA IDENT "DQ RB80"
          68 A5    04    AA                 BICW2   #UCB$M_NOCNVRT,UCB$W_DEVSTS(R5) ;ENABLE LOG TO PHYS CONV.
                                
                   05    09D0  2116 30$:    RSB
                                
```

```
                     09D1  2119          .SBTTL  CONTROLLER INITIALIZATION ROUTINE
                     09D1  2120 ;++
                     09D1  2121 ;
                     09D1  2122 ; FUNCTIONAL DESCRIPTION:
                     09D1  2123 ;
                     09D1  2124 ;       THE CORRECT RB730 CSR ADDRESS IS COMPUTED.
                     09D1  2125 ;
                     09D1  2126 ;       AUTOCONFIGURE UTILIZES A TEMPORARY UNIBUS CSR ADDRESS FOR
                     09D1  2127 ;       CONFIGURING THIS CONTROLLER.  THE TRUE CSR ADDRESS IS LOCATED
                     09D1  2128 ;       EXACTLY ONE PAGE ABOVE THE ADAPTOR CSR (THE DEVICE REGISTERS
                     09D1  2129 ;       ARE ACTUALLY IN THE ADAPTOR CONTROL REGISTER REGION)
                     09D1  2130 ;
                     09D1  2131 ;       THE OPERATING SYSTEM CALLS THIS ROUTINE:
                     09D1  2132 ;               - AT SYSTEM STARTUP
                     09D1  2133 ;               - DURING DRIVER LOADING
                     09D1  2134 ;               - DURING RECOVERY FROM POWER FAILURE
                     09D1  2135 ;
                     09D1  2136 ; INPUTS:
                     09D1  2137 ;
                     09D1  2138 ;       R4      - CSR ADDRESS (DEVICE CONTROL STATUS REGISTER)
                     09D1  2139 ;       R5      - IDB ADDRESS (INTERRUPT DATA BLOCK)
                     09D1  2140 ;       R6      - DDB ADDRESS (DEVICE DATA BLOCK)
                     09D1  2141 ;       R8      - CRB ADDRESS (CHANNEL REQUEST BLOCK)
                     09D1  2142 ;       ALL INTERRUPTS ARE LOCKED OUT
                     09D1  2143 ;
                     09D1  2144 ; OUTPUTS:
                     09D1  2145 ;
                     09D1  2146 ;       IDB$L_CSR       - CORRECT RB730 CSR ADDRESS
                     09D1  2147 ;
                     09D1  2148 ;--
                     09D1  2149
                     09D1  2150 DQ_RB730_INIT:                          ;CONTROLLER INITIALIZATION
                     09D1  2151          .ASSUME ADP$L_CSR EQ 0
          50    14 B5    D0  09D1  2152          MOVL    @IDB$L_ADP(R5),R0          ;FETCH ADAPTOR CSR ADDRESS
     65   0200 C0    DE  09D5  2153          MOVAL   ^X200(R0),IDB$L_CSR(R5)    ;STORE CSR IN IDB
                         05  09DA  2154          RSB
                     09DB  2155
```

```
                        09DB  2157          .SBTTL  UNIT DELIVERY ROUTINE
                        09DB  2158 ;++
                        09DB  2159 ;
                        09DB  2160 ; DQ_DELIVER - UNIT DELIVERY ROUTINE
                        09DB  2161 ;
                        09DB  2162 ; FUNCTIONAL DESCRIPTION:
                        09DB  2163 ;
                        09DB  2164 ; THIS ROUTINE IS CALLED BY AUTOCONFIGURE TO TEST FOR A UNITS
                        09DB  2165 ; PRESCENCE OR ABSCENCE ON THE CONTROLLER
                        09DB  2166 ;
                        09DB  2167 ; INPUTS:
                        09DB  2168 ;
                        09DB  2169 ;       R0-R3    - SCRATCH
                        09DB  2170 ;       R4       - ADDRESS OF ADAPTOR CONFIGURATION REGISTER
                        09DB  2171 ;       R5       - UNIT NUMBER TO BE CONFIGURED
                        09DB  2172 ;       R6       - ADDRESS OF CONFIGURATION CONTROL REGISTER
                        09DB  2173 ;       R7       - ADDRESS OF CONFIGURATION CONTROL BLOCK (ACF BLOCK)
                        09DB  2174 ;       R8       - ADDRESS OF ADAPTOR CONTROL BLOCK
                        09DB  2175 ;
                        09DB  2176 ;       ACF$B_CUNIT(R7) - UNIT NUMBER TO BE TESTED
                        09DB  2177 ;
                        09DB  2178 ; OUTPUTS:
                        09DB  2179 ;
                        09DB  2180 ;       R0       - LBS IF UNIT FOUND, LBC IF NO SUCH UNIT
                        09DB  2181 ;
                        09DB  2182 ;--
                        09DB  2183
                        09DB  2184 DQ_DELIVER:                                  ;UNIT DELIVERY ROUTINE
                  54    DD  09DB  2185          PUSHL   R4                              ;SAVE R4
        54  0200 C6    9E  09DD  2186          MOVAB   ^X200(R6),R4                    ;COMPUTE ADDRESS OF CSR
                  52    D4  09E2  2187          CLRL    R2                              ;PREPARE FOR UNIT NUMBER
52  02  08    55    F0  09E4  2188          INSV    R5,#8,#2,R2                     ;LOAD DRIVE SELECT BITS
              FEF5    30  09E9  2189          BSBW    DQ_WAIT                         ;WAIT FOR CONTROLLER READY
            03 50     E9  09EC  2190          BLBC    R0,50$                          ;BRANCH IF CONTROLLER BUSY
              FECF    30  09EF  2191          BSBW    DQ_GETSTS                       ;ATTEMPT GET STATUS
                        09F2  2192                                                  ;R0=1 IF OK, 0 IF NO UNIT
 64   00000040 8F    CA  09F2  2193 50$:     BICL    #RB_CS_M_IE,RB_CS(R4)           ;DISABLE INTERRUPTS
            54    8E    D0  09F9  2194          MOVL    (SP)+,R4                        ;RESTORE R4
                        05  09FC  2195          RSB                             ;RETURN STATUS TO CALLER
                            09FD  2196
                            09FD  2197
```

```
                     09FD  2199          .SBTTL  REGISTER DUMP ROUTINE
                     09FD  2200 ;++
                     09FD  2201 ;
                     09FD  2202 ; DQ_REGDUMP - REGISTER DUMP ROUTINE
                     09FD  2203 ;
                     09FD  2204 ; FUNCTIONAL DESCRIPTION:
                     09FD  2205 ;
                     09FD  2206 ; THIS ROUTINE IS CALLED TO SAVE THE DEVICE REGISTERS AND UBA RESOURCE
                     09FD  2207 ; REGISTERS IN A SPECIFIED BUFFER. IT IS CALLED FROM THE DEVICE ERROR
                     09FD  2208 ; LOGGING ROUTINE AND FROM THE DIAGNOSTIC BUFFER FILL ROUTINE.
                     09FD  2209 ;
                     09FD  2210 ; INPUTS:
                     09FD  2211 ;
                     09FD  2212 ;       R0      - ADDRESS OF REGISTER SAVE BUFFER
                     09FD  2213 ;       R4      - ADDRESS OF DEVICE CONTROL STATUS REGISTER (CSR)
                     09FD  2214 ;       R5      - ADDRESS OF UNIT CONTROL BLOCK (UCB)
                     09FD  2215 ;
                     09FD  2216 ; OUTPUTS:
                     09FD  2217 ;
                     09FD  2218 ;       THE DEVICE AND UBA REGISTERS ARE SAVED IN THE SPECIFIED BUFFER.
                     09FD  2219 ;       R0 CONTAINS THE ADDRESS OF THE NEXT EMPTY LONGWORD IN THE BUFFER.
                     09FD  2220 ;       ALL REGISTERS EXCEPT R1 AND R2 ARE PRESERVED.
                     09FD  2221 ;
                     09FD  2222 ;--
                     09FD  2223
                     09FD  2224 DQ_REGDUMP:                          ;REGISTER DUMP ROUTINE
     80    0D   D0   09FD  2225          MOVL    #<RB_NUM_REGS+6>,(R0)+  ;INSERT NUMBER OF REGISTERS
51   00CC C5   DE   0A00  2226          MOVAL   UCB$L_DQ_CS(R5),R1      ;GET ADDRESS OF SAVED DEVICE REGISTERS
     52   05   9A   0A05  2227          MOVZBL  #<RB_NUM_REGS-2>,R2     ;GET NUMBER OF DEVICE REGISTERS TO MOVE
     80   81   D0   0A08  2228 10$:     MOVL    (R1)+,(R0)+             ;DUMP REGISTER IN BUFFER
       FA 52   F5   0A0B  2229          SOBGTR  R2,10$                  ;IF GTR - STILL MORE TO MOVF
                    0A0E  2230
80   00C4 C5   3C   0A0E  2231          MOVZWL  UCB$W_EC1(R5),(R0)+     ;ECC POSITION REGISTER
80   00C6 C5   3C   0A13  2232          MOVZWL  UCB$W_EC2(R5),(R0)+     ;ECC PATTERN REGISTER
                    0A18  2233
  52   24 A5   D0   0A18  2234          MOVL    UCB$L_CRB(R5),R2        ;FETCH CRB ADDRESS
  80   37 A2   9A   0A1C  2235          MOVZBL  CRB$L_INTD+VEC$B_DATAPATH(R2),(R0)+ ;DUMP DATAPATH NUMBER
           80   D4   0A20  2236          CLRL    (R0)+                   ;DUMP DATAPATH REGISTER (ALWAYS 0)
      80   81   D0   0A22  2237          MOVL    (R1)+,(R0)+             ;DUMP FINAL MAP REGISTER
      8C   81   D0   0A25  2238          MOVL    (R1)+,(R0)+             ;DUMP PREVIOUS MAP REGISTER
                    0A28  2239
                    0A28  2240          ASSUME  VEC$B_NUMREG EQ VEC$W_MAPREG+2 ;ASSUME START AND NUMBER CONTIG
  80   34 A2   D0   0A28  2241          MOVL    CRB$L_INTD+VEC$W_MAPREG(R2),(R0)+;DUMP MAP REGISTERS
                    0A2C  2242
80   00F6 C5   D0   0A2C  2243          MOVL    UCB$L_DQ_PREVDA(R5),(R0)+ ;DUMP PREVIOUS DISK ADDRESS
                    0A31  2244
                 05 0A31  2245          RSB                             ;RETURN
                    0A32  2246
                    0A32  2247 DQ_END:                              ;ADDRESS OF LAST LOCATION IN DRIVER
                    0A32  2248          .END
```

```
$$$                     = 00000020 R      02
$$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
ADP$L_CSR               = 00000000
APPLY_ECC                 000002A1 R      03
AT$_UBA                 = 00000001
AVAILABLE                 000001E2 R      03
BRW_RETREG                000005B7 R      03
BUG$_UBMAPEXCED           ******** X    03
CDF_AVAILABLE           = 00000011
CDF_DRVCLR              = 00000004
CDF_NOP                 = 00000000
CDF_OFFSET              = 00000006
CDF_PACKACK             = 00000008
CDF_READDATA            = 0000000C
CDF_READHEAD            = 0000000E
CDF_READTRACKD          = 00000010
CDF_RECAL               = 00000003
CDF_RELEASE             = 00000005
CDF_RETCENTER           = 00000007
CDF_SEEK                = 00000002
CDF_STARTSPNDL          = 00000009
CDF_UNLOAD              = 00000001
CDF_WRITECHECK          = 0000000A
CDF_WRITEDATA           = 0000000B
CDF_WRITEHEAD           = 0000000D
CDF_WRITETRACKD         = 0000000F
CHECKECC                  00000235 R      03
CHECKOPI                  000008D3 R      03
CRB$L_INTD              = 00000024
DC$_DISK                = 00000001
DDB$K_PACK              = 00000001
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_NHM               = 00000200
DEV$M_ODV               = 08000000
DEV$M_RND               = 10000000
DEV$M_SHR               = 00010000
DPT$C_LENGTH            = 00000038
DPT$C_VERSION           = 00000004
DPT$INITAB                00000038 R      02
DPT$M_SVP               = 00000002
DPT$REINITAB              0000006D R      02
DPT$TAB                   00000000 R      02
DQ$DDT                    00000000 RG     03
DQ_CLASSIFY               0000097A R      03
DQ_DELIVER                000009DB R      03
DQ_END                    00000A32 R      03
DQ_FUNCTABLE              00000080 R      03
DQ_GETSTS                 000008C1 R      03
DQ_INT                    0000074D R      03
DQ_RB730_INIT             000009D1 R      03
DQ_READHDR                000008A3 R      03
DQ_REGDUMP                000009FD R      03
DQ_REGSAVE                000007E0 R      03
DQ_REI                    00000747 R      03
DQ_RESET                  000008BB R      03
DQ_STARTIO                00000114 R      03
DQ_UNEXINT                0000086C R      03
DQ_UNIT_INIT              0000090D R      03
DQ_WAIT                   000008E1 R      03
DRCLR                     00000398 R      03
DRVCLR                    000001F0 R      03
DT$_RB02                = 00000012
DT$_RB80                = 00000013
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$DEVICTMO              ******** X    03
EXE$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
EXGETSTS                  000008C5 R      03
EX_IMED                   000003B0 R      03
FATAL                     00000744 R      03
FATALERR                  000002C2 R      03
FDISPATCH                 0000018A R      03
FEXL                      0000035A R      03
FTAB                      00000038 R      03
FUNCTAB_LEN             = 00000094
FUNCXT                    00000321 R      03
F_AVAILABLE             = 00000004
F_DRVCLR                = 00000004
F_GETSTATUS             = 00000004
F_NOP                   = 00000000
F_OFFSET                = 00000000
F_PACKACK               = 00000004
F_READDATA              = 0000000C
F_READHEAD              = 00000008
F_READTRACKD            = 00000000
F_RECAL                 = 00000006
F_RELEASE               = 00000000
F_RETCENTER             = 0000C000
F_SEEK                  = 00000006
F_STARTSPNDL            = 00000000
```

```
F_UNLOAD                =  00000004
F_WRITECHECK            =  00000002
F_WRITEDATA             =  0000000A
F_WRITEHEAD             =  00000000
F_WRITETRACKD           =  00000000
IDB$L_ADP               =  00000014
IDB$L_CSR               =  00000000
IDB$L_OWNER             =  00000004
IDB$L_UCBLST            =  00000018
IMMED                     000003AD R      03
IO$M_DATACHECK          =  00004000
IO$V_DATACHECK          =  0000000E
IO$V_INHRETRY           =  0000000F
IO$V_SKPSECINH          =  00000009
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$_READVBLK            =  00000031
IO$_RECAL               =  00000003
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$_WRITEHEAD           =  0000000D
IO$_WRITELBLK           =  00000020
IO$_WRITEPBLK           =  0000000B
IO$_WRITEVBLK           =  00000030
IOC$ALOUBAMAPN            ********  X   03
IOC$APPLYECC              ********  X   03
IOC$DIAGBUFILL            ********  X   03
IOC$LOADUBAMAPA           ********  X   03
IOC$MNTVER                ********  X   03
IOC$MOVTOUSER             ********  X   03
IOC$RELCHAN               ********  X   03
IOC$REQCOM                ********  X   03
IOC$REQDATAPNW            ********  X   03
IOC$REQPCHANL             ********  X   03
IOC$RETURN                ********  X   03
IOC$WFIKPCH               ********  X   03
IPL$_POWER              =  0000001F
IRP$C_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
NOMAPS                    00000517 R      03
NOP                       000001E7 R      03
NORMAL                    00000225 R      03
OFFSET                    000001E7 R      03
PACKACK                   000001D4 R      03
POSIT                     00000421 R      03
PR$_IPL                 =  00000012
PREPROCESS                0000011B R      03
PWRFAIL                   000006A0 R      03
RB_BA                     00000004
RB_BC                     00000008
RB_CMD                    0000001C
RB_CS                     00000000
RB_CS_M_ASSI            =  08000000
RB_CS_M_ATN             =  000F0000
RB_CS_M_CE              =  00008000
RB_CS_M_CRDY            =  00000080
RB_CS_M_DCK             =  00000800
RB_CS_M_DE              =  00004000
RB_CS_M_DLT             =  00001000
RB_CS_M_DRDY            =  00000001
RB_CS_M_FMT             =  20000000
RB_CS_M_IE              =  00000040
RB_CS_M_IR              =  01000000
RB_CS_M_NXM             =  00002000
RB_CS_M_OPI             =  00000400
RB_CS_M_SSE             =  00800000
RB_CS_M_SSEI            =  00400000
RB_CS_M_TYP             =  04000000
RB_CS_S_ATN             =  00000004
RB_CS_S_ECS             =  00000002
RB_CS_S_FCODE           =  00000003
RB_CS_V_ATN             =  00000010
RB_CS_V_CE              =  0000000F
RB_CS_V_DCK             =  0000000B
RB_CS_V_DE              =  0000000E
RB_CS_V_DS              =  00000008
RB_CS_V_ECS             =  00000014
RB_CS_V_FCODE           =  00000001
RB_CS_V_NXM             =  0000000D
RB_CS_V_OPI             =  0000000A
RB_CS_V_SSE             =  00000017
RB_DA                     0000000C
RB_EC1                    00000014
RB_EC2                    00000018
RB_MP                     00000010
RB_MP_C_SLM             -  00000005
RB_MP_M_BH              =  00000008
```

```
RB_MP_M_DSE             =  00000100
RB_MP_M_HCE             =  00004000
RB_MP_M_HO              =  00000010
RB_MP_M_MRK             =  00000001
RB_MP_M_PLGV            =  00000200
RB_MP_M_RST             =  00000008
RB_MP_M_SPD             =  00000800
RB_MP_M_STS             =  00000002
RB_MP_M_VC              =  00000200
RB_MP_M_WDE             =  00008000
RB_MP_M_WGE             =  00000400
RB_MP_V_PLGV            =  00000009
RB_MP_V_VC              =  00000009
RB_MP_V_WGE             =  0000000A
RB_MP_V_WL              =  0000000D
RB_MP_V_WTP             =  0000000D
RB_NUM_REGS             =  00000007
READDATA                   000001FD R      03
READHEAD                   000001F6 R      03
READTRACKD                 000001E7 R      03
RECAL                      000001F0 R      03
RECALB                     000003E6 R      03
RELEASE                    000001E7 R      03
RESETDRIVE                 000002B8 R      03
RETCENTER                  000001E7 R      03
RETHDR                     000005BA R      03
RETREG                     000006B0 R      03
RETRY                      00000728 R      03
RETRYERR                   000002AA R      03
SEEK                       000001F0 R      03
SEEKI                      00000444 R      03
SIZ...                  =  00000020
SPECOND                    00000672 R      03
SS$_CTRLERR             =  00000054
SS$_DATACHECK           =  0000005C
SS$_DRVERR              =  0000008C
SS$_MEDOFL              =  000001A4
SS$_NORMAL              =  00000001
SS$_PARITY              =  000001F4
SS$_TIMEOUT             =  0000022C
SS$_VOLINV              =  00000254
SS$_WASECC              =  00000639
SS$_WRITLCK             =  0000025C
STARTSPNDL                 000001E7 R      03
SUCCESS                    0000073C R      03
TRANSFER                   00000221 R      03
UBI$L_MAP               =  00000800
UCB$B_CEX               =  00000093
UCB$B_DEVCLASS          =  00000040
UCB$B_DEVTYPE           =  00000041
UCB$B_DIPL              =  0000005E
UCB$B_DQ_FLAGS             000000C9
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$K_DQ_LEN            =  000000FA
UCB$K_LCL_DISK_LENGTH   =  000000CC
UCB$L_CRB               =  00000024
UCB$L_DEVCHAR           =  00000038
UCB$L_DEVCHAR2          =  0000003C
UCB$L_DEVDEPEND         =  00000044
UCB$L_DPC               =  0000009C
UCB$L_DQ_BA                000000D0
UCB$L_DQ_BC                000000D4
UCB$L_DQ_CS                000000CC
UCB$L_DQ_CURDA             000000F2
UCB$L_DQ_DA                000000D8
UCB$L_DQ_DPR               000000E8
UCB$L_DQ_FMPR              000000E0
UCB$L_DQ_MP                000000DC
UCB$L_DQ_PMPR              000000E4
UCB$L_DQ_PREVDA            000000F6
UCB$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_DQ_DIP            =  00000002
UCB$M_DQ_ECC_DEFER      =  00000004
UCB$M_DQ_SIP            =  00000001
UCB$M_ECC               =  00000001
UCB$M_NOCNVRT           =  00000004
UCB$M_ONLINE            =  00000010
UCB$M_POWER             =  00000020
UCB$M_TIMOUT            =  00000040
UCB$M_VALID             =  00000800
UCB$V_DIAGBUF           =  00000001
UCB$V_DQ_DIP            =  00000001
UCB$V_DQ_ECC_DEFER      =  00000002
UCB$V_DQ_SIP            =  00000000
UCB$V_ECC               =  00000000
UCB$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$W_DQ_HDR1              000000EC
UCB$W_DQ_HDR2              000000EE
UCB$W_DQ_HDR3              000000F0
UCB$W_ECT               =  000000C4
UCB$W_EC2               =  000000C6
UCB$W_FUNC              =  0000009A
```

```
UCB$W_OFFSET                    = 000000C8
UCB$W_STS                       = 00000064
UCB$W_UNIT                      = 00000054
UNLOAD                            000001E2 R      03
UPDATE                            00000605 R      03
VEC$B_DATAPATH                  = 00000013
VEC$B_NUMREG                    = 00000012
VEC$L_ADP                       = 00000014
VEC$L_IDB                       = 00000008
VEC$L_INITIAL                   = 0000000C
VEC$L_UNITINIT                  = 00000018
VEC$S_MAPREG                    = 0000000F
VEC$V_MAPLOCK                   = 0000000F
VEC$V_MAPREG                    = 00000000
VEC$V_PATHLOCK                  = 00000007
VEC$W_MAPREG                    = 00000010
WRITECHECK                        000001F6 R      03
WRITECHK                          000005EE R      03
WRITEDATA                         000001FD R      03
WRITEHEAD                         000001F0 R      03
WRITETRACKD                       000001E7 R      03
XFER                              000004DA R      03
_TMP$VAL                        = 00000044
```

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
$$$105_PROLOGUE	00000082 (130.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE
$$$115_DRIVER	00000A32 (2610.)	03 (3.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG

Performance indicators !

Phase	Page faults	CPU Time	Elapsed Time
Initialization	31	00:00:00.05	00:00:01.69
Command processing	115	00:00:00.36	00:00:05.99
Pass 1	634	00:00:20.22	00:02:22.64
Symbol table sort	0	00:00:02.57	00:00:14.43
Pass 2	387	00:00:04.98	00:00:31.10
Symbol table output	22	00:00:00.21	00:00:01.35
Psect synopsis output	0	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	1191	00:00:28.41	00:03:17.22

```
The working set limit was 2550 pages.
166368 bytes (325 pages) of virtual memory were used to buffer the intermediate code.
There were 130 pages of symbol table space allocated to hold 2370 non-local and 86 local symbols.
2248 source lines were read in Pass 1, producing 23 object records in Pass 2.
58 pages of virtual memory were used to define 55 macros.
```

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

Macro library name                   Macros defined
------------------                   --------------
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                 33
_$255$DUA28:[SYSLIB]STARLET.MLB;2              11
TOTALS (all libraries)                         44

2514 GETS were required to define 44 macros.

There were no errors, warnings or information messages.

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

DQDRIVER
LIS

DLDRIVER
LIS

DRDRIVER
LIS

DMDRIVER
LIS