```
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
```

FILEID**PASCSCTL

```
PPPPPPPP      AAAAAA      SSSSSSSS    CCCCCCCC    SSSSSSSS    CCCCCCCC  TTTTTTTTTT  LL
PPPPPPPP      AAAAAA      SSSSSSSS    CCCCCCCC    SSSSSSSS    CCCCCCCC  TTTTTTTTTT  LL
PP       PP  AA      AA  SS          CC          SS          CC            TT      LL
PP       PP  AA      AA  SS          CC          SS          CC            TT      LL
PP       PP  AA      AA  SS          CC          SS          CC            TT      LL
PP       PP  AA      AA  SS          CC          SS          CC            TT      LL
PPPPPPPP     AA      AA    SSSSSS    CC            SSSSSS    CC            TT      LL
PPPPPPPP     AA      AA    SSSSSS    CC            SSSSSS    CC            TT      LL
PP           AAAAAAAAAA          SS  CC                  SS  CC            TT      LL
PP           AAAAAAAAAA          SS  CC                  SS  CC            TT      LL
PP           AA      AA          SS  CC                  SS  CC            TT      LL              ....
PP           AA      AA          SS  CC                  SS  CC            TT      LL              ....
PP           AA      AA  SSSSSSSS      CCCCCCCC  SSSSSSSS      CCCCCCCC      TT      LLLLLLLLLL      ....
PP           AA      AA  SSSSSSSS      CCCCCCCC  SSSSSSSS      CCCCCCCC      TT      LLLLLLLLLL      ....

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
```

```
                                                                    L 1
(38)     2102          -       SCS$CLOSE_CDT,  CLEANUP CDT AND COMPLETE
(38)     2103          -                       PENDING CONNX MGMT CALL
(39)     2141          -       SCS$MAP_SCSSTS  MAP SCS STATUS TO VMS STATUS
(40)     2181          -       SCS$MAP_VMSSTS, MAP VMS STATUS TO SCS
(41)     2217      ERROR HANDLING ROUTINES
(41)     2218          -       SCS$DISC_VCFAIL, PROCESS DISCONNECT CALL
(41)     2219          -                        FOR CDT ON FAILING PB
(42)     2278          -       SCS$VC_CLOSED,  HANDLE VC CLOSED ON
(42)     2279          -                       SPECIFIED PATH BLOCK
(43)     2367          -       SCS$NOTIFY_SYSAP, SEARCH CDT LIST AND
(43)     2368          -                         HANDLE CDT'S IN
(43)     2369          -                         VARIOUS STATES
(44)     2530          -       CHK_NO_CDTS,    IF ALL CDTS ON PB DISCONNECTED,
(44)     2531          -                       SEND CACHE CLEAR MSG
(45)     2606          -       SCS$CACHECLR,   PROCESS REC'D MARKER MSG THAT
(45)     2607          -                       SAYS THE PORT CACHE IS CLEAR
(45)     2608          -                       OF TRAFFIC ON SPECIFIED VC
```

```
0000     1         .TITLE  PASCSCTL
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 ;
0000    29 ; FACILITY:
0000    30 ;
0000    31 ;       VAX/VMS EXECUTIVE, I/O DRIVERS
0000    32 ;
0000    33 ; ABSTRACT:  SCS CONTROL INCLUDING CONNECTION MANAGEMENT, PROCESSING
0000    34 ;            RECEIVED CREDIT MESSAGES, AND SHARING OF THE SINGLE
0000    35 ;            SCS SEND BUFFER PER PATH.
0000    36 ;
0000    37 ; AUTHOR:  N. KRONENBERG,  MAY 1981
0000    38 ;
0000    39 ; MODIFIED BY:
0000    40 ;
0000    41 ;       V03-018 NPK3054         N. Kronenberg           24-Jun-1984
0000    42 ;               Change SCS$REQ_SCSSEND to check if a CDT is already
0000    43 ;               waiting for the SCS send buffer before putting it
0000    44 ;               on the queue.  If it is waiting, don't queue, but
0000    45 ;               just update the block state.
0000    46 ;
0000    47 ;       V03-017 NPK3052         N. Kronenberg           4-May-1984
0000    48 ;               Make SCS$CLOSE_CDT a global entry like it was supposed
0000    49 ;               to be in the first place.
0000    50 ;
0000    51 ;       V03-016 NPK3048         N. Kronenberg           4-Apr-1984
0000    52 ;               Modify SCS$VCCLOSED/SCS$SETCRT_CLSD to set aux
0000    53 ;               status to the value in PB$W_VCFAIL_RSN (host shutdown)
0000    54 ;               if there is a nonzero value there.
0000    55 ;
0000    56 ;       V03-015 NPK3046         N. Kronenberg           8-Mar-1984
0000    57 ;               Fix SCS$NOTIFY_SYSAP to completed an outstanding
```

```
0000     58 ;               reject with success and return the cdt to the listening
0000     59 ;               state.  Previously the cdt was closed.
0000     60 ;
0000     61 ;       V03-014 NPK3045         N. Kronenberg           23-Feb-1984
0000     62 ;               Add to SCS$VCCLOSED the instruction to set
0000     63 ;               PB$C_VC_FAIL after checking if a VC failure is already
0000     64 ;               in progress.
0000     65 ;
0000     66 ;       V03-013 NPK3042         N. Kronenberg            6-Feb-1984
0000     67 ;               Modify SCS$VCCLOSED to simply return if a vc closure
0000     68 ;               is already in progress.  Add a new entry to this routine,
0000     69 ;               SCS$SETCKT_CLSD to be called when a SETCKT closed is
0000     70 ;               successfully completed.
0000     71 ;
0000     72 ;       V03-012 NPK3039         N. Kronenberg           11-Jan-1984
0000     73 ;               Repair automatic transition of formative PB to fully
0000     74 ;               open:  Continue to call CNF$SCSMSG_REC, but don't
0000     75 ;               assume that a fully open PB actually resulted. (It
0000     76 ;               would fail, if the ENTER_PB previously failed
0000     77 ;               or now fails either due to lack of pool or because
0000     78 ;               the system has the same system name or system ID, but
0000     79 ;               different incarnation as some system to which we
0000     80 ;               already have circuit(s).)
0000     81 ;
0000     82 ;               If no fully open PB resulted, then the vc will have
0000     83 ;               been closed by CNF$SCSMSG_REC, and the remote system's
0000     84 ;               configuration poller will detect that the circuit is
0000     85 ;               gone and clean up the remote's dangling CONNECT_REQ.
0000     86 ;               The whole problem of a CONNECT_REQ arriving when we
0000     87 ;               have already told the port to open the vc, but when
0000     88 ;               we then failed to do the ENTER_PB could have been
0000     89 ;               avoided by delaying the vc open command to the port
0000     90 ;               till after the ENTER_PB.  This solution was rejected
0000     91 ;               because it would mean that a fast enough remote system
0000     92 ;               might always get its connect request through before
0000     93 ;               the local system could tell its port (after the ENTER_PB)
0000     94 ;               to open the vc.  Consequently, the open command to
0000     95 ;               the port remains as early as possible and is undone
0000     96 ;               later if that is necessary.  This optimizes the common
0000     97 ;               case at the expense of the error case.
0000     98 ;
0000     99 ;               It is possible in the future that sufficiently different
0000    100 ;               cpu speeds may lead to the situation where the fast cpu
0000    101 ;               can always get a connect request sent to a slow system
0000    102 ;               before the slow system can tell its port to open the
0000    103 ;               vc.  If this happens, the current behavior should be
0000    104 ;               reevaluated -- a delay before sending connect requests
0000    105 ;               may be needed.
0000    106 ;
0000    107 ;       V03-011 NPK3037         N. Kronenberg           11-Nov-1983
0000    108 ;               Add optional $DEBUGCHECK on receipt of connect request
0000    109 ;               with no path block in hand or if VC closed or cache
0000    110 ;               clear processing is entered with no path block.
0000    111 ;               Fix SCS$NOTIFY_SYSAP and CHECK_NO_CDTS to suppress
0000    112 ;               send of cache clear if the port is dead.  Also, fix
0000    113 ;               CHECK_NO_CDTS to remove the PB and init the port in
0000    114 ;               the case where the last CDT to be removed did not
```

```
0000   115 ;               result in a call to the SYSAP error routine and
0000   116 ;               a SYSAP disconnect.
0000   117 ;
0000   118 ;       V03-010 ROW0200         Ralph O. Weber          2-AUG-1983
0000   119 ;               Change SCS$ALL_SCSREC error path after INI$ALLOC_MSG to simply
0000   120 ;               RSB.  The previous version was POPing unsaved registers from
0000   121 ;               the stack thus corrupting those registers.  This change relies
0000   122 ;               on EXE$ALONONPAGED returning SS$_INSFMEM.  EXE$ALONONPAGED was
0000   123 ;               recently changed to do this.
0000   124 ;
0000   125 ;       V03-009 NPK3029         N. Kronenberg           18-Jul-1983
0000   126 ;               Enhancements for V4.0:
0000   127 ;               Change to use symbols for SCS status/reason codes.
0000   128 ;               Allow receipt of a connect request to complete a
0000   129 ;               start handshake in case the last ack in the start
0000   130 ;               handshake was missed.
0000   131 ;               Fix disconnect of connection because of vc fail
0000   132 ;               to return both con_rec and rej_sent connections to
0000   133 ;               the listen state.  (Previously rej_sent connections
0000   134 ;               were deleted.)
0000   135 ;
0000   136 ;       V03-008 KTA3046         Kerbey T. Altmann       29-Mar-1983
0000   137 ;               Redo for SCS/PPD split.
0000   138 ;
0000   139 ;       V03-007 NPK3023         N. Kronenberg           1-Mar-1983
0000   140 ;               Check destination connection sequence number on
0000   141 ;               credit requests and responses.
0000   142 ;
0000   143 ;       V03-006 ROW0147         Ralph O. Weber          15-DEC-1982
0000   144 ;               Correcc offset on matching address in SCS$MAP_VMSSTS table
0000   145 ;               lookup.
0000   146 ;
0000   147 ;       V03-005 NPK3008         N. Kronenberg           6-Oct-1982
0000   148 ;               Change SCS status codes to conform to architecture.
0000   149 ;
0000   150 ;       V03-004 ROW0096         Ralph O. Weber          7-JUN-1982
0000   151 ;               Add calls to error logging routines in REC_CON_REQ and
0000   152 ;               BREAK_VC.  Also added necessary reference to $PAERDEF macro.
0000   153 ;               This change will be in a new driver image shipped in V3.1.
0000   154 ;
0000   155 ;       V03-003 NPK2019         N. Kronenberg           6-Apr-1982
0000   156 ;               Change BAD_DATABASE to crash port instead of bugcheck.
0000   157 ;
0000   158 ;       V03-002 NPK2018         N. Kronenberg           25-Mar-1982
0000   159 ;               Added routine SCS$ALLOC_DGPPD to allocate short dgs.
0000   160 ;               Removed test code from SCS$DEAL_DG.
0000   161 ;
0000   162 ;       V03-001 NPK2016         N. Kronenberg           18-Mar-1982
0000   163 ;               Fixed .TITLE
0000   164 ;
0000   165 ;--
```

```
                     .SBTTL  DEFINITIONS

;
; Set PSECT to driver code:
;

                     .PSECT  $$$115_DRIVER,LONG

;
; System definitions (LIB.MLB):
;

                     .nocross
                     $CDLDEF                         ; Connection Descriptor List format
                     $CDTDEF                         ; Connection Descriptor Table format
                     $CDRPDEF                        ; Class Driver Request Pkt format
                     $PBDEF                          ; Path Block format
                     $PDTDEF                         ; Port Descriptor Table
                     $SCSDEF                         ; SCS message format
                     $SSDEF                          ; System service messages

;
; PADRIVER definitions (PALIB.MLB):
;
                     $PAPDTDEF                       ; PA specific PDT extension.
                     $PAERDEF                        ; Port driver error code values
                     .cross
```

```
                        0000    195         .SBTTL  DATA
                        0000    196
                        0000    197 ;+
                        0000    198 ; Tables to convert SCS status codes to corresponding VMS status codes.
                        0000    199 ;-
                        0000    200 ;
                        0000    201 ; SCS status codes:
                        0000    202 ;
                        0000    203
                        0000    204 SCS_STATUS_TAB:
                        0000    205
                        0000    206
                   000A 0000    207         .WORD   SCS$C_STNOMAT           ; No matching listener
                   0012 0002    208         .WORD   SCS$C_STNORS            ; No resources in listener
                   0001 0004    209         .WORD   SCS$C_STNORMAL          ; General success status
                   0019 0006    210         .WORD   SCS$C_STDISC            ; CDT disconnected
                   0021 0008    211         .WORD   SCS$C_STINSFCR          ; Insufficient credits on connect
                   0000 000A    212         .WORD   0                       ; Patch space
                   0000 000C    213         .WORD   0                       ;
                   0000 000E    214         .WORD   0                       ;
                   0000 0010    215         .WORD   0                       ;
                        0012    216
              00000012  0012    217 SCS_STATUS_LEN = .-SCS_STATUS_TAB       ; # bytes in SCS status table
                        0012    218
                        0012    219 ;
                        0012    220 ; VMS status codes corresponding to SCS codes.  This table must
                        0012    221 ; follow above table.
                        0012    222 ;
                        0012    223
                        0012    224 VMS_STATUS_TAB:
                        0012    225
                   215C 0012    226         .WORD   SS$_NOLISTENER          ; No matching listener
                   206C 0014    227         .WORD   SS$_REMRSRC             ; No resources in listener
                   0001 0016    228         .WORD   SS$_NORMAL              ; Success
                   204C 0018    229         .WORD   SS$_DISCONNECT          ; CDT disconnected
                   0001 001A    230         .WORD   SS$_NORMAL              ; Default VMS status for patch space
                   0001 001C    231         .WORD   SS$_NORMAL              ;
                   0001 001E    232         .WORD   SS$_NORMAL              ;
                   0001 0020    233         .WORD   SS$_NORMAL              ;
                        0022    234
                        0022    235 ;
                        0022    236 ; Length of the SCS overhead.
                        0022    237 ;
                        0022    238
                        0022    239 SCS$GL_SCSSIZE::
                        0022    240
              0000000E  0022    241         .LONG   SCS$C_OVHD
                        0026    242
```

```
                     0026   244          .SBTTL  SCS LAYER INITIALIZATION
                     0026   245
                     0026   246 ;+
                     0026   247 ; This routine is called by the INIT module after the PDT has been set
                     0026   248 ; up and befor any port specific hardware initialization.  It gives the
                     0026   249 ; SCS layer a chance to do any initialization it needs.
                     0026   250 ;-
                     0026   251
                     0026   252 SCS$INITIAL::
                     0026   253
FFD7'  31            0026   254          BRW     FPC$INITIAL             ; Do it in FPC for now
                     0029   255
```

```
                     0029   257          .SBTTL  PROCESS REC'D SCS CONTROL MESSAGES
                     0029   258 ;+
                     0029   259 ; This routine is called by module PAINTR when a message is received which is
                     0029   260 ; message type other than SCS$C_APPL_MSG.  This routine dispatches
                     0029   261 ; on the various message types that are possible to routines that
                     0029   262 ; process the received messages.
                     0029   263 ;
                     0029   264 ; Inputs:
                     0029   265 ;
                     0029   266 ;       R2                      -Addr of received message
                     0029   267 ;       R4                      -Addr of PDT
                     0029   268 ;
                     0029   269 ; Outputs:
                     0029   270 ;
                     0029   271 ;       R4                      -Preserved
                     0029   272 ;       other registers         -Destroyed
                     0029   273 ;-
                     0029   274
                     0029   275 ;
                     0029   276 ; Message format assumptions:
                     0029   277 ;
                     0029   278
                     0029   279 ASSUME  SCS$C_CON_REQ   EQ 0
                     0029   280 ASSUME  SCS$C_CON_RSP   EQ 1
                     0029   281 ASSUME  SCS$C_ACCP_REQ  EQ 2
                     0029   282 ASSUME  SCS$C_ACCP_RSP  EQ 3
                     0029   283 ASSUME  SCS$C_REJ_REQ   EQ 4
                     0029   284 ASSUME  SCS$C_REJ_RSP   EQ 5
                     0029   285 ASSUME  SCS$C_DISC_REQ  EQ 6
                     0029   286 ASSUME  SCS$C_DISC_RSP  EQ 7
                     0029   287 ASSUME  SCS$C_CR_REQ    EQ 8
                     0029   288 ASSUME  SCS$C_CR_RSP    EQ 9
                     0029   289
                     0029   290
                     0029   291          .ENABL  LSB
                     0029   292
                     0029   293 SCS$REC_SCSMSG::
                     0029   294
F4 A2    B5          0029   295          TSTW    SCS$W_MTYPE(R2)         ; Connect request?
   07    13          002C   296          BEQL    10$                     ; If so, skip CDT addr calculation
 FFCF'   30          002E   297          BSBW    FPC$CHK_DCONID          ; Check destination CONID
                     0031   298                                          ;  and get CDT addr
01 50    E8          0031   299          BLBS    R0,10$                  ; Branch if good  CONID
         05          0034   300          RSB                             ; Else leave
                     0035   301
                     0035   302 10$:     CASE    SCS$W_MTYPE(R2),<-      ; Dispatch to routine based on msg type:
                     0035   303                  REC_CON_REQ,-           ;  CONNECT_REQ
                     0035   304                  REC_CON_RSP,-           ;  CONNECT_RSP
                     0035   305                  REC_ACCP_REQ,-          ;  ACCEPT_REQ
                     0035   306                  REC_ACCP_RSP,-          ;  ACCEPT_RSP
                     0035   307                  REC_REJ_REQ,-           ;  REJECT_REQ
                     0035   308                  REC_REJ_RSP,-           ;  REJECT_RSP
                     0035   309                  REC_DISC_REQ,-          ;  DISCONNECT_REQ
                     0035   310                  REC_DISC_RSP,-          ;  DISCONNECT_RSP
                     0035   311                  REC_CR_REQ,-            ;  CREDIT_REQ
                     0035   312                  REC_CR_RSP>             ;  CREDIT_RSP
 04DE    31          004E   313          BRW     BREAK_VC                ; Other types are illegal
```

PROCESS REC'D SCS CONTROL MESSAGES

```
0051   314
0051   315
0051   316          .DSABL  LSB
```

```
                     0051   318          .SBTTL  -       REC_CR_REQ,     PROCESS REC'D CREDIT REQUEST
                     0051   319
                     0051   320 ;+
                     0051   321 ; This routine processes a credit request (received separately from
                     0051   322 ; an appliction message).  The algorithm is:
                     0051   323 ;
                     0051   324 ; For positive credit field:
                     0051   325 ;
                     0051   326 ;       CDT$W_SEND(R3) = CDT$W_SEND(R3) + credit
                     0051   327 ;
                     0051   328 ; For negative credit field (credits being retracted):
                     0051   329 ;
                     0051   330 ;       Maximum returnable credit = Maximum of 0 or
                     0051   331 ;                                   (CDT$W_SEND - CDT$W_MIN_SEND)
                     0051   332 ;
                     0051   333 ;       Number to return = Minimum of (Max returnable) or (-credit field)
                     0051   334 ;
                     0051   335 ; Inputs:
                     0051   336 ;
                     0051   337 ;       R2                      -Addr of credit message
                     0051   338 ;       R3                      -Addr of CDT
                     0051   339 ;       R4                      -Addr of PDT
                     0051   340 ;
                     0051   341 ; Outputs:
                     0051   342 ;
                     0051   343 ;       R0,R1                   -Destroyed
                     0051   344 ;       other registers         -Preserved
                     0051   345 ;-
                     0051   346
                     0051   347          .ENABL  LSB
                     0051   348
                     0051   349 REC_CR_REQ:
                     0051   350
      FA A2    B1    0051   351          CMPW    SCS$L_DST_CONID+2(R2),- ; Is the connection seq # in the
      1A A3          0054   352                  CDT$L_LCONID+2(R3)      ;  msg = seq # in cdt?
        54    12     0056   353          BNEQ    FATAL_CR_ERR            ; Branch if not
51    F6 A2    B0    0058   354          MOVW    SCS$W_CREDIT(R2),R1     ; Get # credits extended
        1C    18     005C   355          BGEQ    UPDATE_SEND             ; Branch if positive
      4A A3    A3    005E   356          SUBW3   CDT$W_MINSEND(R3),-     ; Compute # we can return
50    40 A3          0061   357                  CDT$W_SEND(R3),R0       ;
        02    18     0064   358          BGEQ    10$                     ; Branch if not negative
        50    B4     0066   359          CLRW    R0                      ; Else # we can return = 0
                     0068   360
   51    51    AE    0068   361 10$:     MNEGW   R1,R1                   ; Compute -(credit field)
   50    51    B1    006B   362          CMPW    R1,R0                   ; Is it less than, or equal to
                     006E   363                                          ;  maximum returnable?
        03    1B     006E   364          BLEQU   20$                     ; Branch if so
   51    50    B0    0070   365          MOVW    R0,R1                   ; Else choose max returnable
   51    51    AE    0073   366 20$:     MNEGW   R1,R1                   ; Calc -Min(max returnable,-credit)
F6 A2    51    B0    0076   367          MOVW    R1,SCS$W_CREDIT(R2)     ; Tell remote about partial
                     007A   368
                     007A   369 UPDATE_SEND:
                     007A   370
40 A3    51    A0    007A   371          ADDW    R1,CDT$W_SEND(R3)       ; Update our send credit
      0366   30     007E   372          BSBW    SEND_CR_RSP             ; Turn messge around for RSP
                     0081   373
      40 A3    B5    0081   374 30$:     TSTW    CDT$W_SEND(R3)          ; Any send credits now?
```

```
15   13  0084    375          BEQL    40$                         ; Branch if not
            0086    376          $RESUME_FP      -                   ; Else resume
            0086    377                  @CDT$L_CRWAITQFL(R3),-      ;  next waiter
            0086    378                  QEMPTY=40$                  ;  branching if none
E6   11  0099    379          BRB     30$                         ; Check if more credit
            009B    380
     05  009B    381 40$:     RSB                                 ; Return
            009C    382
            009C    383          .DSABL  LSB
```

```
                     009C    385          .SBTTL  -       REC_CR_RSP,     PROCESS REC'D CREDIT RESPONSE
                     009C    386
                     009C    387 ;+
                     009C    388 ; This routine is called upon receipt of a response to a previously
                     009C    389 ; sent CREDIT_REQ.  REC_CR_RSP updates the record of the number
                     009C    390 ; of sent credits held by the remote SCS (CDT$W_REC) and then
                     009C    391 ; resumes the SCS message send process in case any connection
                     009C    392 ; is waiting for the SCS send buffer.
                     009C    393 ;
                     009C    394 ; Inputs:
                     009C    395 ;
                     009C    396 ;       R2                      -Addr of credit response message
                     009C    397 ;       R3                      -Addr of CDT
                     009C    398 ;       R4                      -Addr of PDT
                     009C    399 ;
                     009C    400 ; Outputs:
                     009C    401 ;
                     009C    402 ;       R0-R2                   -Destroyed
                     009C    403 ;       other registers         -Preserved
                     009C    404 ;-
                     009C    405
                     009C    406          .ENABL  LSB
                     009C    407
                     009C    408 REC_CR_RSP:
                     009C    409
        FA A2   B1   009C    410          CMPW    SCS$L_DST_CONID+2(R2),- ; Is connection seq # in msg
        1A A3        009F    411                  CDT$L_LCONID+2(R3)      ;  = seq # in cdt?
           09   12   00A1    412          BNEQ    FATAL_CR_ERR            ; Branch if not
50      F6 A2   B0   00A3    413          MOVW    SCS$W_CREDIT(R2),R0     ; Get credit confirmed
           03   19   00A7    414          BLSS    FATAL_CR_ERR            ; Branch if negative
                     00A9    415                                          ; Negative credit--> remote SCS
                     00A9    416                                          ;  thinks we asked for credit
                     00A9    417                                          ;  back and we never do that.
            02EA    31   00A9    418          BRW     SCS$RESUM_SEND          ; Resume SCS send waiter, if any
                     00AC    419
                     00AC    420 FATAL_CR_ERR:
                     00AC    421
            0480    31   00AC    422          BRW     BREAK_VC                ; Break virtual circuit
                     00AF    423
                     00AF    424          .DSABL  LSB
```

```
                                .SBTTL  -       REC_CON_REQ,    PROCESS REC'D CONNECT REQUEST

                        ;+
                        ; Receipt of a connect request causes the transition of the path block
                        ; associated with this port-port vc from the formative to the open state
                        ; if necessary.  (It is necessary only if the final datagram in the start
                        ; handshake was lost.)
                        ;
                        ; Next, REC_CON_REQ looks up the target fork process in the directory of
                        ; listeners;  if the target is not listening, match failure status is
                        ; returned to the remote.  If the target is listening, the listening CDT
                        ; receives assorted information about the connect requestor, and the
                        ; listening SYSAP is called at its listen message address.
                        ;
                        ; If the target process is busy handling a previous connect request
                        ; (it is in the con_rec or rej_sent state), then 'no resources' status
                        ; is returned to the connect requestor.  The assumption is that, in this
                        ; case, the requestor will reissue the connect request later.
                        ;
                        ; Inputs:
                        ;
                        ;       R2                      -Addr of msg buffer containing CONNECT_REQ
                        ;       R4                      -Addr of PDT
                        ;
                        ; Outputs:
                        ;
                        ;       R0-R2                   -Destroyed
                        ;       other registers         -Preserved
                        ;-


                                .ENABL  LSB

                        REC_CON_REQ:

FF4E'   30                      BSBW    CNF$SCSMSG_REC          ; Complete formative PB transition
                        ;                                       ;  if necessary. (This routine may
                        ;                                       ;  find a formative PB, but may fail
                        ;                                       ;  for a number or reasons.  See
                        ;                                       ;  revision history on this module
                        ;                                       ;  NPK3039)
FF4B'   30                      BSBW    CNF$LKP_PB_MSG2         ; Look up PB given PDT and remote port
04 50   E8                      BLBS    R0,VC_OPEN              ; Branch if found open PB.
                        ;                                       ;  Else, vc is closed and connect
                        ;                                       ;  request arrived during window when
                        ;                                       ;  we had vc open in anticipation of
                        ;                                       ;  a successful handshake.
FF45'   30                      BSBW    INT$INS_MFREEQ          ; Discard CONNECT_REQ to free queue
                        ;                                       ;  -- sender will discover closed vc
                        ;                                       ;  via polling.
        05                      RSB                             ; Return

                        VC_OPEN:

51   04 A2      DE              MOVAL   SCS$T_DST_PROC(R2),R1   ; Get addr of target process name
00000000'GF     16              JSB     G^SCS$LOCLOOKUP         ; Look it up in the directory
     66 50      E9              BLBC    R0,NO_MATCH             ; Branch if failed
```

```
                     00C9   483          $DISPATCH -
                     00C9   484                  CDT$W_STATE(R3),-                ; Dispatch on state:
                     00C9   485                  <-                               ;
                     00C9   486                  <CDT$C_CON_REC, NO_RESOURCE>,-   ;  CONNECT in progress
                     00C9   487                  <CDT$C_REJ_SENT,NO_RESOURCE>,-   ;  CONNECT in progress
                     00C9   488                  <CDT$C_LISTEN,  CON_LISTEN>,-    ;  Just listening
                     00C9   489                  >
                     00E4   490
           0448   31 00E4   491          BRW     BREAK_VC                         ; Illegal state
                     00E7   492
                     00E7   493 CON_LISTEN:
                     00E7   494
             52   DD 00E7   495          PUSHL   R2                               ; Save addr of CONNECT_REQ
           0363   30 00E9   496          BSBW    SCS$ALL_SCSREC                   ; Allocate SCS buffer to listening CDT
             52 8ED0 00EC   497          POPL    R2                               ; Retreive addr of CONNECT_REQ
          42 50   E9 00EF   498          BLBC    R0,NO_RESOURCE                   ; Branch if no pool
           0314   30 00F2   499          BSBW    SCS$COPY_ACCP                    ; Copy info about remote to CDT
           0320   30 00F5   500          BSBW    SCS$SAVE_REQ                     ; Copy connect request to listener
                     00F8   501                                                   ;  SCS buffer.
           FF05'  30 00F8   502          BSBW    CNF$LKP_PB_MSG2                  ; Look up path block given PDT and
          3C 50   E9 00FB   503          BLBC    R0,BAD_DATABASE                  ; Branch if couldn't find PB --
                     00FE   504                                                   ;  shouldn't get seg msg unless
                     00FE   505                                                   ;  there is a PB (VC open)
      1C A3   51   D0 00FE   506          MOVL    R1,CDT$L_PB(R3)                  ; Save PB addr in listen CDT
           0C A1   D0 0102   507          MOVL    PB$B_RSTATION(R1),-              ; Also save remote station addr,
           20 A3              0105   508                  CDT$B_RSTATION(R3)
           10 A1   B0 0107   509          MOVW    PB$B_RSTATION+4(R1),-
           24 A3              010A   510                  CDT$B_RSTATION+4(R3)
      10 A3   54   D0 010C   511          MOVL    R4,CDT$L_PDT(R3)                 ;  and PDT addr in listen CDT
              09   B0 0110   512          MOVW    #CDT$C_CON_REC,-                 ; Move connection state from
           28 A3              0112   513                  CDT$W_STATE(R3)          ;  listen to connect received
           34 A1   D0 0114   514          MOVL    PB$L_CDTLST(R1),-                ; Insert listening CDT on PB list
           6C A3              0117   515                  CDT$L_CDTLST(R3)         ;  of CDT's in case of VC or
      34 A1   53   D0 0119   516          MOVL    R3,PB$L_CDTLST(R1)               ;  power failure
           50   01   B0 011D   517          MOVW    #SCS$C_STNORMAL,R0               ; Set status to match successful
             0296   30 0120   518          BSBW    SCS$SEND_RSP                     ; Generate response
      52   2C A3   D0 0123   519          MOVL    CDT$L_SCSMSG(R3),R2              ; Get addr of copy of msg
      52   04 A2   9E 0127   520          MOVAB   SCS$T_DST_PROC(R2),R2            ; Compute addr of part SYSAP wants
           00 B3   16 012B   521          JSB     @CDT$L_MSGINPUT(R3)              ; Call listener input addr
                   05 012E   522          RSB                                      ; Return to PAINIT to dequeue
                     012F   523                                                   ;  more responses
                     012F   524
                     012F   525 NO_MATCH:
                     012F   526
           50   0A   B0 012F   527          MOVW    #SCS$C_STNOMAT,R0                ; Set status to no matching process
                03   11 0132   528          BRB     10$                              ; Join common code
                     0134   529
                     0134   530 NO_RESOURCE:
                     0134   531
           50   12   B0 0134   532          MOVW    #SCS$C_STNORS,R0                 ; Set status to no resources
                     0137   533
              027F   31 0137   534 10$:     BRW     SCS$SEND_RSP                     ; Send response
                     013A   535
                     013A   536 BAD_DATABASE:
                     013A   537
                     013A   538          $DEBUGCHECK #ERR$V_DEB_NOPB      ; Optionally, bugcheck on this error
50   8004 8F   32 014D   539          CVTWL   #<PAER$K_ES_CNPB ! ^X8000>, R0   ; Log received connect request
```

```
FEAB'  30  0152   540          BSBW    ELOG$PACKET                     ; from port w/o PB error.
FEA8'  30  0155   541          BSBW    INT$INS_MFREEQ          ; Put SCS receive buffer back
           0158   542                                          ;  on free queue
FEA5'  30  0158   543          BSBW    ERR$CRASH_PORT          ; Init port crash
       05  015B   544          RSB                             ; Return to interrupt service
           015C   545
           015C   546          .DSABL  LSB
```

```
                     548          .SBTTL  -       REC_CON_RSP,    PROCESS REC'D CONNECT RESPONSE
                     549
                     550 ;+
                     551 ; This routine is called upon receipt of a CONNECT_RSP.  Action 's as
                     552 ; follows:
                     553 ;
                     554 ;       CONNECT_RSP STATUS      CDT STATE       ACTION
                     555 ;
                     556 ;       Match                   CON_SENT        Move state to CON_ACK.
                     557 ;                                               Resume SCS send wait queue.
                     558 ;
                     559 ;       No match,               CON_SENT        Deallocate CDT and all its
                     560 ;       No resources                            buffers.  Resume SYSAP with
                     561 ;                                               error status.  Resume SCS
                     562 ;                                               send wait queue.
                     563 ;
                     564 ; Inputs:
                     565 ;
                     566 ;       R2                      -Addr of CONNECT_RSP message
                     567 ;       R3                      -Addr of CDT
                     568 ;       R4                      -Addr of PDT
                     569 ;
                     570 ; Outputs:
                     571 ;
                     572 ;       R0-R2                   -Destroyed
                     573 ;       other registers         -Preserved (if connect acknowleged)
                     574 ;       R3                      -Destroyed if connect not acknowleged
                     575
                     576 ;-
                     577
                     578          .ENABL  LSB
                     579
                     580 REC_CON_RSP:
                     581
                     582          $CHK_CDTSTATE -                    ; Verify CDT state is
                     583                  CON_SENT,-                 ;  connect sent;  if not,
                     584                  ERROR=BREAK_VC             ;  fatal state error
      02 A2    B1    0165   585          CMPW    SCS$W_STATUS(R2),- ; Was connect acknowleged?
         01          0168   586                  #SCS$C_STNORMAL    ;
         07    12    0169   587          BNEQ    CONNECT_FAIL       ; Branch if not
         08    B0    016B   588          MOVW    #CDT$C_CON_ACK,-   ; Move CDT state to connect ack'ed
      28 A3          016D   589                  CDT$W_STATE(R3)    ;
       0224    31    016F   590          BRW     SCS$RESUM_SEND     ; Resume SCS send queue
                     0172   591
                     0172   592 CONNECT_FAIL:
                     0172   593
       037D    30    0172   594          BSBW    SCS$DEAL_ALLBUF    ; Deallocate all CDT's buffers
       0428    30    0175   595          BSBW    SCS$MAP_SCSSTS     ; Map SCS status to VMS
         50    DD    0178   596          PUSHL   R0                 ;  and save on stack
       0219    30    017A   597          BSBW    SCS$RESUM_SEND     ; Resume SCS send queue
   55  68 A3    D0    017D   598          MOVL    CDT$L_FR5(R3),R5   ; Restore SYSAP's context: R5
      64 A3    DD    0181   599          PUSHL   CDT$L_FPC(R3)      ;  and PC from CDT
00000000'GF    16    0184   600          JSB     G^SCS$DEALL_CDT    ; Deallocate CDT too
   50  04 AE    D0    018A   601          MOVL    4(SP),R0           ; Retreive VMS status
   04 AE    54    D0    018E   602          MOVL    R4,4(SP)           ; Save PDT addr over status
         9E    16    0192   603          JSB     @(SP)+             ; Call SYSAP with error status
            54 8ED0    0194   604          POPL    R4                 ; Retreive PDT address
```

```
05  0197   605          RSB                             ; Return
    0198   606
    0198   607          .DSABL  LSB
```

```
                     0198   609          .SBTTL  -       REC_ACCP_REQ,   PROCESS REC'D ACCEPT REQUEST
                     0198   610
                     0198   611 ;+
                     0198   612 ; this routine is called upon receipt or an ACCEPT_REQ message.  Action
                     0198   613 ; is as follows:
                     0198   614 ;
                     0198   615 ;       CDT State               Action
                     0198   616 ;
                     0198   617 ;       CON_ACK                 Copy accept message and parameters
                     0198   618 ;                               to CDT;  Set CDT state open.
                     0198   619 ;                               Send ACCEPT_RSP.  Resume SYSAP
                     0198   620 ;                               with success status.
                     0198   621 ;
                     0198   622 ;       Sequence # check        Send back a response that tells
                     0198   623 ;       fails                   remote SCS this CDT went away.
                     0198   624 ;
                     0198   625 ; Inputs:
                     0198   626 ;
                     0198   627 ;       R2                      -Addr of CONNECT_RSP message
                     0198   628 ;       R3                      -Addr of CDT
                     0198   629 ;       R4                      -Addr of PDT
                     0198   630 ;
                     0198   631 ; Outputs:
                     0198   632 ;
                     0198   633 ;       R0-R2                   -Destroyed
                     0198   634 ;       other registers         -Preserved
                     0198   635 ;-
                     0198   636
                     0198   637          .ENABL  LSB
                     0198   638
                     0198   639 REC_ACCP_REQ:
                     0198   640
      03DA     30    0198   641          BSBW    SCS$CHK_SEQNUM          ; Check that this is our CDT
    27 50      E9    019B   642          BLBC    R0,CDT_CLOSED           ; Branch if not-- we must have
                     019E   643                                          ;  done a unilateral disconnect
                     019E   644          $CHK_CDTSTATE   -               ; Verify that the CDT state is
                     019E   645                  CON_ACK,-               ;  CONNECT ack'ed;  if not,
                     019E   646                  ERROR=BREAK_VC          ;  fatal state error
      026E     30    01A7   647          BSBW    SCS$SAVE_REQ            ; Save copy of ACCEPT_REQ message
      025C     30    01AA   648          BSBW    SCS$COPY_ACCP           ; Copy info from ACCEPT_REQ msg
                     01AD   649                                          ;  to local CDT
        02     B0    01AD   650          MOVW    #CDT$C_OPEN,-           ; Set CDT state open
    28 A3            01AF   651                  CDT$W_STATE(R3)         ;
    50 01      B0    01B1   652          MOVW    #SCS$C_STNORMAL,R0      ; Set to send success response
      0202     30    01B4   653          BSBW    SCS$SEND_RSP            ; Send ACCEPT response to remote
    50 01      3C    01B7   654          MOVZWL  #SS$_NORMAL,R0          ; Set to return success
52  2C A3      D0    01BA   655          MOVL    CDT$L_SCSMSG(R3),R2     ; Get addr of success message
52  04 A2      DE    01BE   656          MOVAL   SCS$T_DST_PROC(R2),R2   ; Step R2 past SCS header
      03A3     31    01C2   657          BRW     RESUME_CONCALL          ; Resume SYSAP
                     01C5   658
                     01C5   659 CDT_CLOSED:
                     01C5   660
    50 19      B0    01C5   661          MOVW    #SCS$C_STDISC,R0        ; Set response status to error
      01EE     31    01C8   662          BRW     SCS$SEND_RSP            ;  and send the response
                     01CB   663
                     01CB   664          .DSABL  LSB
```

```
                     01CB   666          .SBTTL  -       REC_ACCP_RSP,   PROCESS REC'D ACCEPT RESPONSE
                     01CB   667
                     01CB   668 ;+
                     01CB   669 ; This routine is called upon receipt of an ACCEPT_RSP message.  Action
                     01CB   670 ; is as follows:
                     01CB   671 ;
                     01CB   672 ;       CDT State               Action
                     01CB   673 ;
                     01CB   674 ;       ACCP_SENT               If the ACCEPT_RSP has success status,
                     01CB   675 ;                               set CDT state open.  Resume SCS
                     01CB   676 ;                               send wait queue.  Resume SYSAP.
                     01CB   677 ;                               If the ACCEPT_RSP has fail status,
                     01CB   678 ;                               then clean up the CDT and return
                     01CB   679 ;                               the RSP error status to the SYSAP.
                     01CB   680 ;
                     01CB   681 ; Inputs:
                     01CB   682 ;
                     01CB   683 ;       R2                      -Addr of ACCEPT_RSP message
                     01CB   684 ;       R3                      -Addr of CDT
                     01CB   685 ;       R4                      -Addr of PDT
                     01CB   686 ;
                     01CB   687 ; Outputs:
                     01CB   688 ;
                     01CB   689 ;       R0-R2                   -Destroyed
                     01CB   690 ;       other registers         -Preserved
                     01CB   691 ;-
                     01CB   692
                     01CB   693          .ENABL  LSB
                     01CB   694
                     01CB   695 REC_ACCP_RSP:
                     01CB   696
                     01CB   697          $CHK_CDTSTATE -                 ; Verify that CDT state is
                     01CB   698                  ACCP_SENT,-             ;  accept sent;  if not,
                     01CB   699                  ERROR=BREAK_VC          ;  fatal state error
          02    B0   01D4   700          MOVW    #CDT$C_OPEN,-           ; Move state to open
      28 A3         01D6   701                  CDT$W_STATE(R3)         ;
50    2C A3    D0   01D8   702          MOVL    CDT$L_SCSMSG(R3),R0     ; Get CONNECT_REQ msg buffer
          53    DD   01DC   703          PUSHL   R3                      ; Save addr of accepting CDT
53    F8 A0    D0   01DE   704          MOVL    SCS$L_DST_CONID(R0),R3  ; Get addr of listening CDT
                     01E2   705                                          ;  saved by ACCEPT call
       035A    30   01E2   706          BSBW    SCS$FREE_LISTEN         ; Put listener back in listen state
          53 8ED0   01E5   707          POPL    R3                      ; Retreive addr of accepting CDT
       03B5    30   01E8   708          BSBW    SCS$MAP_SCSSTS          ; Map SCS status to VMS
          50    DD   01EB   709          PUSHL   R0                      ; Save translated status
       01A6    30   01ED   710          BSBW    SCS$RESUM_SEND          ; Resume SCS send wait queue
          50 8ED0   01F0   711          POPL    R0                      ; Restore saved VMS status
01        50    B1   01F3   712          CMPW    R0,#SS$_NORMAL          ; Was ACCEPT ok?
          03    12   01F6   713          BNEQ    ACCP_ABORT              ; Branch if not
       036D    31   01F8   714          BRW     RESUME_CONCALL          ; Resume SYSAP
                     01FB   715
                     01FB   716 ACCP_ABORT:
                     01FB   717
       0383    31   01FB   718          BRW     SCS$CLOSE_CDT           ; Clean up the CDT that went
                     01FE   719                                          ;  with the ACCEPT request and
                     01FE   720                                          ;  notify caller that ACCEPT failed
                     01FE   721
                     01FE   722          .DSABL  LSB
```

```
                     01FE   724          .SBTTL  -       REC_REJ_REQ,    PROCESS REC'D REJECT REQUEST
                     01FE   725
                     01FE   726 ;+
                     01FE   727 ; This routine is called upon receipt of a REJECT_REQ message.  Action
                     01FE   728 ; is as follows:
                     01FE   729 ;
                     01FE   730 ;       CDT State               Action
                     01FE   731 ;
                     01FE   732 ;       CON_ACK                 Save reject reason and send REJECT_RSP.
                     01FE   733 ;                               Deallocate all buffers owned by CDT.
                     01FE   734 ;                               Deallocate CDT.  Resume SYSAP with
                     01FE   735 ;                               error status and reject reason.
                     01FE   736 ;
                     01FE   737 ;       Sequence # check        Send back a response that tells
                     01FE   738 ;       fails                   remote SCS that this CDT went away
                     01FE   739 ;                               via a unilateral DISCONNECT
                     01FE   740 ;
                     01FE   741 ; Inputs:
                     01FE   742 ;
                     01FE   743 ;       R2                      -Addr of REJECT_REQ message
                     01FE   744 ;       R3                      -Addr of CDT
                     01FE   745 ;       R4                      -Addr of PDT
                     01FE   746 ;
                     01FE   747 ; Outputs:
                     01FE   748 ;
                     01FE   749 ;       R0-R2                   -Destroyed
                     01FE   750 ;       other registers         -Preserved
                     01FE   751 ;-
                     01FE   752
                     01FE   753          .ENABL  LSB
                     01FE   754
                     01FE   755 REC_REJ_REQ:
                     01FE   756
          0374   30  01FE   757          BSBW    SCS$CHK_SEQNUM          ; Check if CDT is ours
        C1 50   E9   0201   758          BLBC    R0,CDT_CLOSED           ; Branch if not -- we must have
                     0204   759                                          ;  done a unilateral DISCONNECT
                     0204   760          $CHK_CDTSTATE   -               ; Verify that CDT state is
                     0204   761                  CON_ACK,-               ;  CONNECT ack'ed;  if not,
                     0204   762                  ERROR=BREAK_VC          ;  fatal state error
          0390   30  020D   763          BSBW    SCS$MAP_SCSSTS          ; Translate SCS reason code to VMS
            50   DD  0210   764          PUSHL   R0                      ; Save reject reason
    55    68 A3   D0  0212   765          MOVL    CDT$L_FR5(R3),R5        ; Restore SYSAP's context: R5,
          64 A3   DD  0216   766          PUSHL   CDT$L_FPC(R3)           ;  and PC from CDT
       50   01   3C  0219   767          MOVZWL  #SCS$C_STNORMAL,R0      ; Set for success status in RSP
          019A   30  021C   768          BSBW    SCS$SEND_RSP            ; Send reject response
          02D0   30  021F   769          BSBW    SCS$DEAL_ALLBUF         ; Deallocate all buffers owned by CDT
 00000000'GF   16  0222   770          JSB     G^SCS$DEALL_CDT         ; Deallocate the CDT
   51   04 AE   D0  0228   771          MOVL    4(SP),R1                ; Get reject reason again
50    0294 8F   3C  022C   772          MOVZWL  #SS$_REJECT,R0          ; Set main status = reject
  04 AE    54   D0  0231   773          MOVL    R4,4(SP)                ; Save PDT addr over reject reason
           9E   16  0235   774          JSB     @(SP)+                  ; Call SYSAP with error status
        54 8ED0   0237   775          POPL    R4                      ; Retreive PDT addr
             05   023A   776          RSB                             ; Return
                  023B   777
                  023B   778          .DSABL  LSB
```

```
                     023B   780          .SBTTL  -       REC_REJ_RSP,    PROCESS REC'D REJECT RESPONSE
                     023B   781
                     023B   782 ;+
                     023B   783 ; This routine is called upon receipt of a REJECT_RSP message.  Action
                     023B   784 ; is as follows:
                     023B   785 ;
                     023B   786 ;       CDT State (listener)            Action
                     023B   787 ;
                     023B   788 ;       CON_REC                         Return state to listen.  Resume
                     023B   789 ;                                       SCS send wait queue.  Resume
                     023B   790 ;                                       SYSAP with success regardless of the
                     023B   791 ;                                       status in the REJECT_RSP.
                     023B   792 ;
                     023B   793 ; Inputs:
                     023B   794 ;
                     023B   795 ;       R2                      -Addr of REJECT_RSP message
                     023B   796 ;       R3                      -Addr of CDT
                     023B   797 ;       R4                      -Addr of PDT
                     023B   798 ;
                     023B   799 ; Outputs:
                     023B   800 ;
                     023B   801 ;       R0-R2                   -Destroyed
                     023B   802 ;       other registers         -Preserved
                     023B   803 ;-
                     023B   804
                     023B   805          .ENABL  LSB
                     023B   806
                     023B   807 REC_REJ_RSP:
                     023B   808
                     023B   809          $CHK_CDTSTATE -                 ; Verify that CDT state is
                     023B   810                  REJ_SENT,-              ;  is connect received;  if not,
                     023B   811                  ERROR=BREAK_VC          ;  fatal state error
          02D6   30  0244   812          BSBW    SCS$DEAL_SCSREC         ; Deallocate the SCS receive buffer
          02F5   30  0247   813          BSBW    SCS$FREE_LISTEN         ; Put listener back in listen state
          0149   30  024A   814          BSBW    SCS$RESUM_SEND          ; Resume SCS send wait queue
    50     01    3C  024D   815          MOVZWL  #SS$_NORMAL,R0          ; Set status to success
          0315   31  0250   816          BRW     RESUME_CONCALL          ; Resume SYSAP
                     0253   817
                     0253   818          .DSABL  LSB
```

```
                     820          .SBTTL  -       REC_DISC_REQ,   PROCESS REC'D DISCONNECT REQUEST
                     821
                     822 ;+
                     823 ; This routine is called upon receipt or a DISCONNECT_REQ message.  It is
                     824 ; valid for several CDT states and action is as follows:
                     825 ;
                     826 ;       CDT State                       Action
                     827 ;
                     828 ;       OPEN                            Set state to disconnect received.
                     829 ;                                       Save disconnect reason.  Send
                     830 ;                                       DISCONNECT_RSP.  Call SYSAP error
                     831 ;                                       address with error status and reason
                     832 ;                                       code.
                     833 ;
                     834 ;       DISC_ACK                        This is a matching DISCONNECT.
                     835 ;                                       Send RSP.  Retreive saved SYSAP
                     836 ;                                       context from CDT.  Deallocate
                     837 ;                                       CDT and receive buffers.  Complete
                     838 ;                                       SYSAP's DISCONNECT call with
                     839 ;                                       success status.
                     840 ;
                     841 ;       DISC_SENT                       This is a matching DISCONNECT
                     842 ;                                       that crossed ours in the mail.
                     843 ;                                       Send RSP and move CDT state to
                     844 ;                                       DISC_MTCH.
                     845 ;
                     846 ;       Sequence # check                Send RSP with success status.
                     847 ;       fails
                     848 ;
                     849 ; Inputs:
                     850 ;
                     851 ;       R2                      -Addr of DISCONNECT_REQ message
                     852 ;       R3                      -Addr of CDT
                     853 ;       R4                      -Addr of PDT
                     854 ;
                     855 ; Outputs:
                     856 ;
                     857 ;       R0-R2                   -Destroyed
                     858 ;       other registers         -Preserved
                     859 ;-
                     860
                     861          .ENABL  LSB
                     862
                     863 REC_DISC_REQ:
                     864
 031F   30           865          BSBW    SCS$CHK_SEQNUM          ; Verify that the CDT is still ours
35 50   E9           866          BLBC    R0,REC_DISC_CLOSED      ; Branch if not-- must have done a
                     867                                          ;  unilateral DISCONNECT
                     868          $DISPATCH -
                     869                  CDT$W_STATE(R3),-               ; Dispatch on current state
                     870                  <-                              ;
                     871                  <CDT$C_OPEN,    REC_DISC_OPEN>,-;  OPEN,
                     872                  <CDT$C_DISC_ACK,REC_DISC_ACK>,- ;  DISCONNECT ack'ed,
                     873                  <CDT$C_DISC_SENT,REC_DISC_SENT>,- ;  DISCONNECT sent
                     874                  >
                     875
 02C6   31           876          BRW     BREAK_VC                        ; Fatal error if other state
```

```
                     0269   877
                     0269   878 REC_DISC_OPEN:
                     0269   879
          04   B0    0269   880          MOVW    #CDT$C_DISC_REC,-       ; Move state to disconnect received
    28 A3             026B   881                  CDT$W_STATE(R3)         ;
     0330    30      026D   882          BSBW    SCS$MAP_SCSSTS          ; Convert SCS status to VMS
       50    DD      0270   883          PUSHL   R0                      ;  and save it
      0144   30      0272   884          BSBW    SCS$SEND_RSP            ; Send DISCONNECT response
       51 8ED0       0275   885          POPL    R1                      ; Restore disconnect reason
50  204C 8F   3C      0278   886          MOVZWL  #SS$_DISCONNECT,R0      ; Set status to disconnected
       54    DD      027D   887          PUSHL   R4                      ; Save R4
    0C B3    16      027F   888          JSB     @CDT$L_ERRADDR(R3)      ; Call SYSAP error addr
       54 8ED0       0282   889          POPL    R4                      ; Restore R4
          05         0285   890          RSB                             ; Return
                     0286   891
                     0286   892 REC_DISC_ACK:
                     0286   893
50        19   B0    0286   894          MOVW    #SCS$C_STDISC,R0        ; Set response status
      012D   30      0289   895          BSBW    SCS$SEND_RSP            ; Send response to DISCONNECT
        27   11      028C   896          BRB     10$                     ; Complete SYSAP DISCONNECT call
                     028E   897
                     028E   898 REC_DISC_CLOSED:
                     028E   899
50        19   B0    028E   900          MOVW    #SCS$C_STDISC,R0        ; Set response status
      0125   31      0291   901          BRW     SCS$SEND_RSP            ; Send response to DISCONNECT
                     0294   902
                     0294   903 REC_DISC_SENT:
                     0294   904
50        01   B0    0294   905          MOVW    #SCS$C_STNORMAL,R0      ; Set response status
      011F   30      0297   906          BSBW    SCS$SEND_RSP            ; Send response
        06   B0      029A   907          MOVW    #CDT$C_DISC_MTCH,-      ; Move CDT state to matching
    28 A3             029C   908                  CDT$W_STATE(R3)         ;  DISCONNECT received
          05         029E   909          RSB                             ; Return
```

```
                     .SBTTL  -       REC_DISC_RSP,   PROCESS REC'D DISCONNECT RESPONSE

                     ;+
                     ; This routine is called upon receipt of a DISCONNECT_RSP.  It is
                     ; valid in more than one state and action is as follows:
                     ;
                     ;       CDT State                       Action
                     ;
                     ;       DISC_SENT                       This is a response to a
                     ;                                       previously sent DISCONNECT.
                     ;                                       Move CDT state to DISC_ACK
                     ;                                       and exit.
                     ;
                     ;       DISC_MTCH                       This is a response to a
                     ;                                       DISCONNECT sent that crossed
                     ;                                       a received DISCONNECT in the
                     ;                                       mail.  It is time to close the
                     ;                                       CDT and notify the SYSAP that
                     ;                                       its DISCONNECT is done.
                     ;
                     ; Inputs:
                     ;
                     ;       R2                      -Addr of DISCONNECT_RSP message
                     ;       R3                      -Addr of CDT
                     ;       R4                      -Addr of PDT
                     ;
                     ; Outputs:
                     ;
                     ;       R0-R2                   -Destroyed
                     ;       other registers         -Preserved
                     ;-

                     REC_DISC_RSP:

                             $DISPATCH -
                                     CDT$W_STATE(R3),-               ; Check for legal state:
                                     <-                              ;
                                     <CDT$C_DISC_SENT,RSP_DISC_SENT>,- ;  DISCONNECT sent,
                                     <CDT$C_DISC_MTCH,RSP_DISC_MTCH>,- ;  Matching DISC rec'd
                                     >

     0284    31              BRW     BREAK_VC                ; Other state is fatal

                     RSP_DISC_SENT:

       03    B0              MOVW    #CDT$C_DISC_ACK,-       ; Move CDT state to
   28 A3                             CDT$W_STATE(R3)         ;  DISCONNECT ack'ed
      00E4    31              BRW     SCS$RESUM_SEND          ; Resume anyone waiting
                                                              ;  for SCS send buffer

                     RSP_DISC_MTCH:

      00E1    30              BSBW    SCS$RESUM_SEND          ; Resume anyone waiting
                                                              ;  for SCS send buffer
50     01    3C      10$:    MOVZWL  #SS$_NORMAL,R0          ; Set status to success
      02C6    31              BRW     SCS$CLOSE_CDT           ;  and complete SYSAP DISCONNECT call
```

```
02BB   968          .DSABL  LSB
```

```
                        .SBTTL  CONNECTION AND SCS CONTROL SUBROUTINES
                        .SBTTL  -       SCS$REQ_SCSSEND, REQUEST SEND OF SCS CONTROL
                        .SBTTL  -                       MESSAGE

                ;+
                ; SCS$REQ_SCSSEND is called when an SCS control message is to be sent.  The
                ; block state defines the kind of control message to send:  CREDIT_REQ,
                ; CONNECT_REQ, REJECT_REQ, ACCEPT_REQ, DISC_REQ, or final
                ; CREDIT followed by DISCONNECT.  The CDT contains sufficient
                ; information to format the outgoing control message.
                ;
                ; Since a single SCS message buffer is available for each path to a
                ; remote system, SYSAP's may have to wait for the buffer to become available.
                ; The wait queue is a linked list of CDT's (linked via CDT$L_WAITQFL/BL) with
                ; listhead in the path block.  A CDT is removed from the wait queue as
                ; soon as it receives control of the SCS send message.  The SCS message
                ; address in the path block is used to determine if the send message
                ; buffer is currently available. (0/non-0 --> not avail/avail.)
                ;
                ; Only one block state may be in effect at a time for a CDT and therefore
                ; the CDT will never need to appear more than once on the wait queue.
                ; However, there are two cases where SCS$REQ_SCSSEND may be called for a
                ; CDT that is already on the SCS send wait queue.  One case is where
                ; two msgs have been queued for receive, each triggers a CREDIT_REQ
                ; (likely if SCSFLOWCUSH and/or MIN_REC are greater than 0), and both
                ; must wait.  The second wait occurs when the CDT is already queued and
                ; waiting.  The second case is when a SYSAP issues a DISCONNECT on a
                ; CDT which already waiting to send a credit.  In all cases, if the CDT
                ; is already waiting to send an SCS control message, it is okay to just
                ; change the block state to the new state as the CDT sits on the queue.
                ; The original request is then replaced by the new request.
                ;
                ; If the SCS send buffer is free, then the control message is sent immediately.
                ; Otherwise, the CDT block state is checked.  If zero, the CDT is inserted
                ; on the tail of the queue.  The CDT block state is then set.
                ;
                ; Inputs:
                ;
                ;       R0                      -Block state code (l.o. 16 bits)
                ;       R3                      -Addr of CDT
                ;       R4                      -Addr of PDT
                ;
                ; Outputs:
                ;
                ;       R0-R2                   -Destroyed
                ;       other registers         -Preserved
                ;-

                        .ENABL  LSB

                SCS$REQ_SCSSEND::

51   1C A3   D0  02BB  MOVL    CDT$L_PB(R3),R1          ; Get path block addr
52   40 A1   D0  02BF  MOVL    PB$L_SCSMSG(R1),R2       ; Get addr of SCS msg buffer
        0F   12  02C3  BNEQ    SEND_NEXT_SEND           ; Branch if buffer available
     2A A3   B5  02C5  TSTW    CDT$W_BLKSTATE(R3)       ; CDT currently waiting to send?
        05   12  02C8  BNEQ    10$                      ; Branch if so
```

```
                30 A3   0E  02CA  1027          INSQUE  CDT$L_WAITQFL(R3),-     ; Link CDT to end of
                3C B1       02CD  1028                  @PB$L_WAITQBL(R1)       ;  SCS send wait queue
                            02CF  1029
            2A A3   50  B0  02CF  1030 10$:     MOVW    R0,CDT$W_BLKSTATE(R3)   ; Save block state
                        05  02D3  1031          RSB                             ; Return to caller (msg not
                            02D4  1032                                          ;  yet sent)
                            02D4  1033
                            02D4  1034 ;
                            02D4  1035 ; The SEND_NEXT_SEND entry is called when the SCS send message buffer
                            02D4  1036 ; is available to send a new control message.  The inputs to this entry:
                            02D4  1037 ;
                            02D4  1038 ;       R0                          -Block state
                            02D4  1039 ;       R1                          -Addr of PB
                            02D4  1040 ;       R2                          -Addr of SCS message buffer
                            02D4  1041 ;       R3                          -Addr of CDT
                            02D4  1042 ;       R4                          -Addr of PDT
                            02D4  1043 ;
                            02D4  1044
                            02D4  1045 ;
                            02D4  1046 ; Block state value assumptions:
                            02D4  1047 ;
                            02D4  1048
                            02D4  1049 ASSUME  CDT$C_CON_PEND  EQ 1
                            02D4  1050 ASSUME  CDT$C_ACCP_PEND EQ 2
                            02D4  1051 ASSUME  CDT$C_REJ_PEND  EQ 3
                            02D4  1052 ASSUME  CDT$C_DISC_PEND EQ 4
                            02D4  1053 ASSUME  CDT$C_CR_PEND   EQ 5
                            02D4  1054 ASSUME  CDT$C_DCR_PEND  EQ 6
                            02D4  1055
                            02D4  1056 SEND_NEXT_SEND:
                            02D4  1057
            40 A1   D4      02D4  1058          CLRL    PB$L_SCSMSG(R1)         ; Show SCS send buffer unavail
                            02D7  1059          CASE    R0,<=                   ; Dispatch on block state:
                            02D7  1060                  BLK_ST_ERR,-            ;  undefined
                            02D7  1061                  SEND_CONNECT,-          ;  Send CONNECT_REQ
                            02D7  1062                  SEND_ACCEPT,-           ;  Send ACCEPT_REQ
                            02D7  1063                  SEND_REJECT,-           ;  Send REJECT_REQ
                            02D7  1064                  SEND_DISCONNECT,-       ;  Send DISCONNECT_REQ
                            02D7  1065                  SEND_CREDIT,-           ;  Send CREDIT_REQ
                            02D7  1066                  SEND_CREDIT>            ;  Send CREDIT_REQ+DISCONNECT_REQ
                            02E9  1067
                            02E9  1068 BLK_ST_ERR:
                            02E9  1069
                            02E9  1070          BUGCHECK CIPORT                 ; Inconsistent CDT state field
                            02F0  1071
                            02F0  1072          .DSABL  LSB
```

```
                     02F0  1074          .SBTTL  -       SEND_CREDIT,    SEND A CREDIT REQUEST
                     02F0  1075
                     02F0  1076 ;+
                     02F0  1077 ; SEND_CREDIT -- Format and send a credit message.
                     02F0  1078 ;
                     02F0  1079 ; Inputs:
                     02F0  1080 ;
                     02F0  1081 ;       R0                      -Block state
                     02F0  1082 ;       R1                      -Addr of PB
                     02F0  1083 ;       R2                      -Addr of message buffer
                     02F0  1084 ;       R3                      -Addr of CDT
                     02F0  1085 ;       R4                      -Addr of PDT
                     02F0  1086 ;
                     02F0  1087 ; Outputs:
                     02F0  1088 ;
                     02F0  1089 ;       R0,R1                   -Destroyed
                     02F0  1090 ;       other registers         -Preserved
                     02F0  1091 ;-
                     02F0  1092
                     02F0  1093 ;
                     02F0  1094 ; Message format and CDT offset assumptions:
                     02F0  1095 ;
                     02F0  1096
                     02F0  1097 ASSUME  SCS$L_DST_CONID+4 EQ SCS$L_SRC_CONID
                     02F0  1098 ASSUME  CDT$L_RCONID+4    EQ CDT$L_LCONID
                     02F0  1099
                     02F0  1100          .ENABL  LSB
                     02F0  1101
                     02F0  1102 SEND_CREDIT:
                     02F0  1103
06   50    B1        02F0  1104          CMPW    R0,#CDT$C_DCR_PEND      ; Is this part of DISCONNECT?
     09    12        02F3  1105          BNEQ    10$                     ; Branch if not
  30 A3    0E        02F5  1106          INSQUE  CDT$L_WAITQFL(R3),-     ; Else requeue CDT for
  3C B1              02F8  1107                  @PB$L_WAITQBL(R1)       ;  DISCONNECT
     04    B0        02FA  1108          MOVW    #CDT$C_DISC_PEND,-      ;  and move CDT state to
  2A A3              02FC  1109                  CDT$W_BLKSTATE(R3)      ;  DISCONNECT pending
                     02FE  1110
     12    B0        02FE  1111 10$:     MOVW    #SCS$C_CR_REQL,-        ; Set length of CREDIT_REQ msg
  F0 A2              0300  1112                  SCS$W_LENGTH(R2)        ;
     08    B0        0302  1113          MOVW    #SCS$C_CR_REQ,-         ; Set SCS message type
  F4 A2              0304  1114                  SCS$W_MTYPE(R2)         ;
                     0306  1115
                     0306  1116 FMT_SCS_CREDIT:                         ; Entry for finishing SCS msg w/credit
                     0306  1117
  46 A3    B0        0306  1118          MOVW    CDT$W_PENDREC(R3),-     ; Copy pending receive
  F6 A2              0309  1119                  SCS$W_CREDIT(R2)        ;  credit to message
  46 A3    A0        030B  1120          ADDW    CDT$W_PENDREC(R3),-     ; Move pending recv credit to
  42 A3              030E  1121                  CDT$W_REC(R3)           ;  receive credit
  46 A3    B4        0310  1122          CLRW    CDT$W_PENDREC(R3)       ; Pending recv credit --> 0
                     0313  1123
                     0313  1124 FMT_SCS:                                ; Entry for finishing SCS w.o. credit
                     0313  1125
  14 A3    7D        0313  1126          MOVQ    CDT$L_RCONID(R3),-      ; Set source connection ID
  F8 A2              0316  1127                  SCS$L_DST_CONID(R2)     ;  and remote connection ID
     2F    11        0318  1128          BRB     SNDMSG                  ; Send message and return msg
                     031A  1129                                          ;  buffer to free queue for
                     031A  1130                                          ;  corresponding RSP
```

```
                        ;
                        ; SEND_CREDIT, SEND A CREDIT REQUEST
                        ;
                        ;+
                        ; SEND_CREDIT - SEND A CREDIT REQUEST
                        ;
                        ; THIS ROUTINE SENDS A CREDIT REQUEST TO THE REMOTE SYSTEM.
                        ;
                        ; INPUTS:
                        ;
                        ;       R5      = ADDRESS OF LSB
                        ;
                        ; OUTPUTS:
                        ;
                        ;       R0      = SS$_NORMAL
                        ;
                        ; SIDE EFFECTS:
                        ;
                        ;       NONE
                        ;-
                        ;
                        .DSABL  LSB
```

```
                        .SBTTL  -       SEND_CONNECT,   SEND A CONNECT_REQ

                ;+
                ; SEND_CONNECT -- Format and send a CONNECT_REQ message.
                ; COMMON_CONNECT -- Complete formatting and transmission of a CONNECT_REQ
                ;                   or ACCEPT_REQ message
                ;
                ; Inputs:
                ;
                ;       R1                      -Addr of PB
                ;       R2                      -Addr of start of message buffer
                ;       R3                      -Addr of CDT
                ;       R4                      -Addr of PDT
                ;
                ;       Connecting/Accepting CDT:
                ;
                ;       CDT$L_RPROCNAM          -Addr of remote proc name
                ;             LPROCNAM          -Addr of local proc name
                ;             CONDAT            -Addr of connect data from local
                ;             INITREC           -Initial credit to extend
                ;             RCONID            -Remote connection ID
                ;             LCONID            -Local connection ID
                ;             MINSEND           -Minimum send credit req'd by local SYSAP
                ;             PDT               -PDT
                ;             SCSMSG            -Addr of message buffer holding CONNECT_REQ
                ;
                ; Outputs:
                ;
                ;       R0,R1                   -Destroyed
                ;       other registers         -Preserved
                ;-
                ;
                ;
                ; Message and data structure adjacency assumptions.  These also apply
                ; to the following routine, SEND_ACCEPT.
                ;

                ASSUME  SCS$W_MTYPE+2   EQ SCS$W_CREDIT
                ASSUME  SCS$W_CREDIT+2  EQ SCS$L_DST_CONID
                ASSUME  SCS$L_DST_CONID+4 EQ SCS$L_SRC_CONID
                ASSUME  SCS$L_SRC_CONID+4 EQ SCS$W_MIN_CR
                ASSUME  SCS$W_MIN_CR+2  EQ SCS$W_STATUS
                ASSUME  SCS$W_STATUS+2  EQ SCS$T_DST_PROC
                ASSUME  SCS$T_DST_PROC+16 EQ SCS$T_SRC_PROC
                ASSUME  SCS$T_SRC_PROC+16 EQ SCS$B_CON_DAT

                ASSUME  CDT$L_RCONID+4  EQ CDT$L_LCONID

                ASSUME  CDT$L_RPROCNAM+4 EQ CDT$L_LPROCNAM
                ASSUME  CDT$L_LPROCNAM+4 EQ CDT$L_CONDAT

                        .ENABL  LSB

                SEND_CONNECT:

0042 8F   B0    MOVW    #SCS$C_CON_REQL,-       ; Set length of CONNECT_REQ msg
  F0 A2         SCS$W_LENGTH(R2)        ;
```

```
                     50   F4 A2    3E  0320  1191          MOVAW   SCS$W_MTYPE(R2),R0      ; Get addr of SCS msg type code
                        8C   00    B0  0324  1192          MOVW    #SCS$C_CON_REQ,(R0)+    ; Set type code
                                       0327  1193
                                       0327  1194 COMMON_CONNECT:
                                       0327  1195
                     80   46 A3    B0  0327  1196          MOVW    CDT$W_PENDREC(R3),(R0)+  ; Copy initial recv credit to msg
                          46 A3    B0  032B  1197          MOVW    CDT$W_PENDREC(R3),-      ; Copy initial to current recv credit
                          42 A3        032E  1198                  CDT$W_REC(R3)            ;  tally in CDT
                          46 A3    B4  0330  1199          CLRW    CDT$W_PENDREC(R3)        ; Zero pending receive
                     80   14 A3    7D  0333  1200          MOVQ    CDT$L_RCONID(R3),(R0)+   ; Copy remote (0) & local connx ID's
                     80   4A A3    3C  0337  1201          MOVZWL  CDT$W_MINSEND(R3),(R0)+  ; Copy min send credit req'd
                     54   50 A3    DE  033B  1202          MOVAL   CDT$L_RPROCNAM(R3),R4    ; Get addr of destination process name
                              11    10  033F  1203          BSBB    MOV_PROCDATA             ; Copy dest process name to msg
                              0F    10  0341  1204          BSBB    MOV_PROCDATA             ;  Also, source process name,
                              0D    10  0343  1205          BSBB    MOV_PROCDATA             ;  and connect data.
                     54   10 A3    D0  0345  1206          MOVL    CDT$L_PDT(R3),R4         ; Restore R4
                          50        D4  0349  1207 SNDMSG: CLRL    R0                       ; Send msg and return buffer to
                     51   1C A3    D0  034B  1208          MOVL    CDT$L_PB(R3),R1          ; Get PB addr in R1
                           FCAE'    31  034F  1209          BRW     INT$SNDMSG               ;  free queue for corresponding response
                                       0352  1210
                                       0352  1211
                                       0352  1212 ;
                                       0352  1213 ; Subroutine to move 16 bytes of data from source address pointed
                                       0352  1214 ; to by R4 to destination address in R0.
                                       0352  1215 ;
                                       0352  1216
                                       0352  1217 MOV_PROCDATA:
                                       0352  1218
                        51   84    D0  0352  1219          MOVL    (R4)+,R1                 ; Get addr of source of data
                        80   81    7D  0355  1220          MOVQ    (R1)+,(R0)+              ; Copy 16 bytes from source to
                        80   81    7D  0358  1221          MOVQ    (R1)+,(R0)+              ;  message buffer
                                   05  035B  1222          RSB                              ; Return
                                       035C  1223
                                       035C  1224          .DSABL  LSB
```

```
                                .SBTTL  -       SEND_ACCEPT,    SEND AN ACCEPT_REQ

                                ;+
                                ; SEND_ACCEPT -- Format and send an ACCEPT_REQ
                                ;
                                ; Inputs:
                                ;
                                ;       R1                      -Addr of PB
                                ;       R2                      -Addr of start of message buffer
                                ;       R3                      -Addr of CDT
                                ;       R4                      -Addr of PDT
                                ;
                                ;       Accepting CDT:
                                ;
                                ;       CDT$L_RPROCNAM          -Addr of remote proc name
                                ;             LPROCNAM          -Addr of local proc name
                                ;             CONDAT            -Addr of connect data from local
                                ;             INITREC           -Initial credit to extend
                                ;             RCONID            -Remote connection ID
                                ;             LCONID            -Local connection ID
                                ;             MINSEND           -Minimum send credit req'd by local SYSAP
                                ;             PDT               -PDT
                                ;
                                ; Outputs:
                                ;
                                ;       R0,R1                   -Destroyed
                                ;       other registers         -Preserved
                                ;-
                                .ENABL  LSB

                                SEND_ACCEPT:

0042 8F   B0                            MOVW    #SCS$C_ACCP_REQL,-      ; Set length of ACCEPT_REQ msg
  F0 A2                                 SCS$W_LENGTH(R2)        ;
50   F4 A2   3E                         MOVAW   SCS$W_MTYPE(R2),R0      ; Get addr of SCS msg type
80     02   B0                         MOVW    #SCS$C_ACCP_REQ,(R0)+   ; Set SCS msg type
       BC   11                          BRB     COMMON_CONNECT          ; Join code common with CONNECT

                                .DSABL  LSB
```

```
                     036B  1267          .SBTTL  -       SEND_REJECT,    SEND A REJECT_REQ
                     036B  1268
                     036B  1269 ;+
                     036B  1270 ; SEND_REJECT -- Send a REJECT_REQ message
                     036B  1271 ;
                     036B  1272 ; Inputs:
                     036B  1273 ;
                     036B  1274 ;       R1                      -Addr of PB
                     036B  1275 ;       R2                      -Addr of start of message buffer
                     036B  1276 ;       R3                      -Addr of CDT
                     036B  1277 ;       R4                      -Addr of PDT
                     036B  1278 ;
                     036B  1279 ; Outputs:
                     036B  1280 ;
                     036B  1281 ;       R0,R1                   -Destroyed
                     036B  1282 ;       other registers         -Preserved
                     036B  1283 ;-
                     036B  1284
                     036B  1285          .ENABL  LSB
                     036B  1286
                     036B  1287 SEND_REJECT:
                     036B  1288
    12   B0  036B  1289          MOVW    #SCS$C_REJ_REQL,-       ; Set REJECT_REQ msg length
F0 A2            036D  1290                  SCS$W_LENGTH(R2)        ;
    04   3C  036F  1291          MOVZWL  #SCS$C_REJ_REQ,-        ; Set SCS msg type
F4 A2            0371  1292                  SCS$W_MTYPE(R2)         ;  and zero credit field
    62   B4  0373  1293          CLRW    SCS$W_MIN_CR(R2)        ; Zero minimum credit field
26 A3    B0  0375  1294          MOVW    CDT$W_REASON(R3),-      ; Copy reject reason
02 A2            0378  1295                  SCS$W_STATUS(R2)        ;  to status
 FF96    31  037A  1296          BRW     FMT_SCS                 ; Finish up and send msg
                     037D  1297
                     037D  1298          .DSABL  LSB
```

```
                     037D  1300          .SBTTL  -       SEND_DISCONNECT, SEND A DISCONNECT_REQ
                     037D  1301
                     037D  1302 ;+
                     037D  1303 ; SEND_DISCONNECT -- Format and send (at low priority) a DISCONNECT_REQ.
                     037D  1304 ;
                     037D  1305 ; Inputs:
                     037D  1306 ;
                     037D  1307 ;       R1                      -Addr of PB
                     037D  1308 ;       R2                      -Addr of start of message buffer
                     037D  1309 ;       R3                      -Addr of CDT
                     037D  1310 ;       R4                      -Addr of PDT
                     037D  1311 ;
                     037D  1312 ; Outputs:
                     037D  1313 ;
                     037D  1314 ;       R0,R1                   -Destroyed
                     037D  1315 ;       other registers         -Preserved
                     037D  1316 ;-
                     037D  1317
                     037D  1318 ;
                     037D  1319 ; Message and CDT format assumptions:
                     037D  1320 ;
                     037D  1321
                     037D  1322 ASSUME  SCS$W_MTYPE+2   EQ SCS$W_CREDIT
                     037D  1323 ASSUME  SCS$L_DST_CONID+4 EQ SCS$L_SRC_CONID
                     037D  1324
                     037D  1325 ASSUME  CDT$L_RCONID+4  EQ CDT$L_LCONID
                     037D  1326
                     037D  1327          .ENABL  LSB
                     037D  1328
                     037D  1329 SEND_DISCONNECT:
                     037D  1330
     12    B0        037D  1331          MOVW    #SCS$C_DISC_REQL,-      ; Set DISCONNECT_REQ length
F0   A2              037F  1332                  SCS$W_LENGTH(R2)        ;
     06    3C        0381  1333          MOVZWL  #SCS$C_DISC_REQ,-       ;  and msg type;  and
F4   A2              0383  1334                  SCS$W_MTYPE(R2)         ;  zero credit field
     62    B4        0385  1335          CLRW    SCS$W_MIN_CR(R2)        ; Zero minimum credit
26   A3    B0        0387  1336          MOVW    CDT$W_REASON(R3),-      ; Copy saved DISCONNECT
02   A2              038A  1337                  SCS$W_STATUS(R2)        ;  reason to status
14   A3    7D        038C  1338          MOVQ    CDT$L_RCONID(R3),-      ; Copy dest, src connx ID's
F8   A2              038F  1339                  SCS$L_DST_CONID(R2)     ;
     50    D4        0391  1340          CLRL    R0                      ; RETFLAG = 0
 FC6A'     31        0393  1341          BRW     INT$SNDMSGL             ; Send off message at low
                     0396  1342 ;                PRIORITY=LOW            ;  priority behind all other
                     0396  1343                                          ;  currently queued msgs and
                     0396  1344                                          ;  block transfers
                     0396  1345
                     0396  1346          .DSABL  LSB
```

```
                     0396  1348          .SBTTL  -       SCS$RESUM_SEND, RESUME SCS CONTROL MESSAGE
                     0396  1349          .SBTTL  -                       SEND QUEUE
                     0396  1350
                     0396  1351 ;+
                     0396  1352 ; This routine is called when an SCS control message response is received.
                     0396  1353 ; Receipt of the response indicates that the SCS send buffer is now free
                     0396  1354 ; and a new SCS control message may be sent to the remote system.  The
                     0396  1355 ; response message buffer is used for the next send.
                     0396  1356 ;
                     0396  1357 ; The wait queue (listhead PB$L_WAITQFL) is checked for waiting CDT's.
                     0396  1358 ; If any CDT is waiting, then it is dequeued and its SCS control message
                     0396  1359 ; sent.  If no CDT is waiting, then the buffer containing the response
                     0396  1360 ; is linked to the path block for future use.
                     0396  1361 ;
                     0396  1362 ; Inputs:
                     0396  1363 ;
                     0396  1364 ;       R2                      -Addr of SCS response message buffer
                     0396  1365 ;       R3                      -Addr of CDT receiving SCS RSP message
                     0396  1366 ;       R4                      -Addr of PDT
                     0396  1367 ;
                     0396  1368 ; Outputs:
                     0396  1369 ;
                     0396  1370 ;       R0-R2                   -Destroyed
                     0396  1371 ;       other registers         -Preserved
                     0396  1372 ;-
                     0396  1373
                     0396  1374          .ENABL  LSB
                     0396  1375
                     0396  1376 SCS$RESUM_SEND:
                     0396  1377
51   1C A3    D0     0396  1378          MOVL    CDT$L_PB(R3),R1         ; Get path block addr
       53    DD     039A  1379          PUSHL   R3                      ; Save CDT receiving response
53   38 B1    0F     039C  1380          REMQUE  @PB$L_WAITQFL(R1),R3    ; Get CDT waiting to send
          0F    1D  03A0  1381          BVS     20$                     ; Branch if nobody waiting
   53    30    C2  03A2  1382          SUBL    #CDT$L_WAITQFL,R3       ; Back up to start of CDT
50   2A A3    B0     03A5  1383          MOVW    CDT$W_BLKSTATE(R3),R0   ; Get block state
     2A A3    B4     03A9  1384          CLRW    CDT$W_BLKSTATE(R3)      ; Zero block state of resumed CDT
       FF25    30  03AC  1385          BSBW    SEND_NEXT_SEND          ; Go send the message
          04    11  03AF  1386          BRB     30$                     ; Join common exit
                     03B1  1387
40 A1    52    D0     03B1  1388 20$:    MOVL    R2,PB$L_SCSMSG(R1)      ; Save SCS msg addr for future
                     03B5  1389
       53 8ED0     03B5  1390 30$:    POPL    R3                      ; Restore CDT receiving response
             05    03B8  1391          RSB                             ; Return
                     03B9  1392
                     03B9  1393          .DSABL  LSB
```

```
                        .SBTTL  -       SCS$SEND_RSP,   TURN SCS REQ AROUND INTO
                        .SBTTL  -                       SCS RESPONSE

                ;+
                ; One receive message buffer per port-port virtual circuit is reserved
                ; for receipt of SCS REQ control messages by all connections on the
                ; VC.  This routine is called with the reserved receive buffer
                ; occupied by an SCS REQ.  SCS$SEND_RSP modified the REQ type message
                ; into its corresponding RSP and sends it to the remote system.  The
                ; RSP message is sent with RETFLAG false so that the message buffer is
                ; returned to the free queue to receive a new SCS REQ.
                ;
                ; Note that certain SCS commands must be retained by the caller
                ; for further processing by the SYSAP.  These are CONNECT_REQ and
                ; ACCEPT_REQ.  the caller is expected om these cases to have copied
                ; the REQ message into the SCS buffer allocated for each CDT
                ; (CDT$L_SCSMSG).
                ;
                ; Inputs:
                ;
                ;       R0                      -Status to send out with RSP, 0 if none
                ;       R1                      -Credit to confirm if applicable
                ;       R2                      -Addr of SCS REQ message
                ;       R4                      -Addr of PDT
                ;
                ; Outputs:
                ;
                ;       R0,R1                   -Destroyed
                ;       other registers         -Preserved
                ;-

                ;
                ; Message format assumptions:
                ;
                ASSUME  SCS$C_CON_REQ   EQ 0
                ASSUME  SCS$C_CON_RSP   EQ 1
                ASSUME  SCS$C_ACCP_REQ  EQ 2
                ASSUME  SCS$C_ACCP_RSP  EQ 3
                ASSUME  SCS$C_REJ_REQ   EQ 4
                ASSUME  SCS$C_REJ_RSP   EQ 5
                ASSUME  SCS$C_DISC_REQ  EQ 6
                ASSUME  SCS$C_DISC_RSP  EQ 7
                ASSUME  SCS$C_CR_REQ    EQ 8
                ASSUME  SCS$C_CR_PSP    EQ 9

                ASSUME  SCS$C_REJ_RSPL  EQ SCS$C_DISC_RSPL

                        .ENABL  LSB

                SCS$SEND_RSP:

                        CASE    SCS$W_MTYPE(R2),<-      ; Dispatch on request type
                                SEND_CON_RSP,-          ;  CONNECT_REQ
                                ILLMSGTYP,-             ;  CONNECT_RSP illegal
                                SEND_ACCP_RSP,-         ;  ACCEPT_REQ
                                ILLMSGTYP,-             ;  ACCEPT_RSP illegal
```

```
                03B9  1452                SEND_REJ_RSP,-          ;  REJECT_REQ
                03B9  1453                ILLMSGTYP,-             ;  REJECT_RSP illegal
                03B9  1454                SEND_DISC_RSP>          ;  DISCONNECT_REQ
                03CC  1455
                03CC  1456 ILLMSGTYP:
                03CC  1457
                03CC  1458          BUGCHECK  CIPORT              ; Any other type is illegal
                03D3  1459
                03D3  1460 SEND_ACCP_RSP:
                03D3  1461
    12    B0    03D3  1462          MOVW      #SCS$C_ACCP_RSPL,-  ; Set response message length
F0  A2          03D5  1463                    SCS$W_LENGTH(R2)    ;
    27    11    03D7  1464          BRB       COMMON_RSP2         ; Join common formatting
                03D9  1465
                03D9  1466 SEND_REJ_RSP:
                03D9  1467 SEND_DISC_RSP:
                03D9  1468
    0E    B0    03D9  1469          MOVW      #SCS$C_REJ_RSPL,-   ; Set response message length
F0  A2          03DB  1470                    SCS$W_LENGTH(R2)    ;
    21    11    03DD  1471          BRB       COMMON_RSP2         ; Join common formatting
                03DF  1472
                03DF  1473 SEND_CON_RSP:
                03DF  1474
    12    B0    03DF  1475          MOVW      #SCS$C_CON_RSPL,-   ; Set CONNECT_RSP length
F0  A2          03E1  1476                    SCS$W_LENGTH(R2)    ;
    62    B4    03E3  1477          CLRW      SCS$W_MIN_CR(R2)    ; Zero minimum credit
    19    11    03E5  1478          BRB       COMMON_RSP2         ; Join common code
                03E7  1479
                03E7  1480 SEND_CR_RSP:
                03E7  1481
    0E    B0    03E7  1482          MOVW      #SCS$C_CR_RSPL,-    ; Set credit RSP length
F0  A2          03E9  1483                    SCS$W_LENGTH(R2)    ;
                03EB  1484
                03EB  1485 COMMON_RSP1:
                03EB  1486
    F4  A2  B6  03EB  1487          INCW      SCS$W_MTYPE(R2)     ; Convert type from REQ to RSP
50  F8  A2  D0  03EE  1488          MOVL      SCS$L_DST_CONID(R2),R0  ; Swap destination
    FC  A2  D0  03F2  1489          MOVL      SCS$L_SRC_CONID(R2),-   ;  and
    F8  A2      03F5  1490                    SCS$L_DST_CONID(R2)     ;  source
FC  A2  50  D0  03F7  1491          MOVL      R0,SCS$L_SRC_CONID(R2)  ;  connection ID's
        50  D4  03FB  1492          CLRL      R0                  ; RETFLAG = false
      FC00' 31  03FD  1493          BRW       INT$TRNMSG
                0400  1494 ;        $TURNMSG RETFLAG=FALSE        ; Send off message
                0400  1495
                0400  1496 COMMON_RSP2:
                0400  1497
    F6  A2  B4  0400  1498          CLRW      SCS$W_CREDIT(R2)    ; Credit = 0
02  A2  50  B0  0403  1499          MOVW      R0,SCS$W_STATUS(R2) ; Status = caller-specified value
        E2  11  0407  1500          BRB       COMMON_RSP1         ; Join common completion
                0409  1501
                0409  1502          .DSABL    LSB
```

```
                     0409  1504          .SBTTL  -       SCS$COPY_ACCP, COPY CONNECTION PARAMETERS
                     0409  1505          .SBTTL  -                      FROM ACCEPT MESSAGE TO CDT
                     0409  1506
                     0409  1507 ;+
                     0409  1508 ; SCS$COPY_ACCP copies the credit, sender's connect ID, and minimum
                     0409  1509 ; credit from the CONNECT_REQ or ACCEPT_REQ message to the SEND credit,
                     0409  1510 ; RCONID, and MINSEND fields of the CDT.
                     0409  1511 ;
                     0409  1512 ; Inputs:
                     0409  1513 ;
                     0409  1514 ;       R2                      -Addr of message contianing CONNECT_REQ
                     0409  1515 ;                                or ACCEPT_REQ
                     0409  1516 ;       R3                      -Addr of CDT
                     0409  1517 ;
                     0409  1518 ; Outputs:
                     0409  1519 ;
                     0409  1520 ;       All registers           -Preserved
                     0409  1521 ;
                     0409  1522 ;-
                     0409  1523
                     0409  1524          .ENABL  LSB
                     0409  1525
                     0409  1526 SCS$COPY_ACCP::
                     0409  1527
F6 A2    B0          0409  1528          MOVW    SCS$W_CREDIT(R2),-      ; Copy extended credit to CDT
40 A3                040C  1529                  CDT$W_SEND(R3)          ;
FC A2    D0          040E  1530          MOVL    SCS$L_SRC_CONID(R2),-   ; Copy remote CONID to CDT
14 A3                0411  1531                  CDT$L_RCONID(R3)        ;
   62    B0          0413  1532          MOVW    SCS$W_MIN_CR(R2),-      ; Copy remote SYSAP's minimum
44 A3                0415  1533                  CDT$W_MINREC(R3)        ;  send credit to CDT
         05          0417  1534          RSB                             ; Return
                     0418  1535
                     0418  1536          .DSABL  LSB
```

```
                     0418  1538          .SBTTL  -       SCS$SAVE_REQ,   COPY SCS REQUEST TO SECOND
                     0418  1539          .SBTTL  -                       MESSAGE BUFFER
                     0418  1540
                     0418  1541 ;+
                     0418  1542 ; SCS$SAVE_REQ copies a received SCS request to an SCS message buffer
                     0418  1543 ; attached to a CDT.  This routine is called when a copy of the request
                     0418  1544 ; must be retained while the initial message is turned around into an
                     0418  1545 ; SCS response.
                     0418  1546 ;
                     0418  1547 ; Inputs:
                     0418  1548 ;
                     0418  1549 ;       R2                      -Addr of SCS buffer containing request
                     0418  1550 ;       R3                      -Addr of CDT
                     0418  1551 ;       CDT$L_SCSMSG            -Addr of CDT's SCS msg buffer
                     0418  1552 ;
                     0418  1553 ; Outputs:
                     0418  1554 ;
                     0418  1555 ;       R0,R1                   -Destroyed
                     0418  1556 ;       Other registers         -Preserved
                     0418  1557 ;-
                     0418  1558
                     0418  1559          .ENABL  LSB
                     0418  1560
                     0418  1561 SCS$SAVE_REQ:
                     0418  1562
          3C   BB    0418  1563          PUSHR   #^M<R2,R3,R4,R5>        ; Save registers
    53  2C A3   D0    041A  1564          MOVL    CDT$L_SCSMSG(R3),R3     ; Pick up receiving buffer
      0042 8F   28    041E  1565          MOVC3   #SCS$C_CON_REQL, -
                     0422  1566                   SCS$W_MTYPE-2(R2),-
F2 A3   F2 A2        0422  1567                   SCS$W_MTYPE-2(R3)       ; Copy message
          3C   BA    0426  1568          POPR    #^M<R2,R3,R4,R5>        ; Restore registers
               05    0428  1569          RSB                             ; Return
                     0429  1570
                     0429  1571          .DSABL  LSB
```

```
                     0429  1573          .SBTTL  MESSAGE AND DATAGRAM BUFFER ALLOCATION
                     0429  1574          .SBTTL  -       SCS$ALL_ALLBUF, ALLOCATE ALL BUFFERS NEEDED
                     0429  1575          .SBTTL  -                       FOR A NEW CONNECTION
                     0429  1576
                     0429  1577 ;+
                     0429  1578 ; SCS$ALL_ALLBUF allocated from nonpaged pool:
                     0429  1579 ;
                     0429  1580 ;       1 SCS buffer for connect request storage, linked to CDT$L_SCSMSG
                     0429  1581 ;       Initial receive credit worth of message buffers
                     0429  1582 ;       Initial dg receive count of datagram buffers.
                     0429  1583 ;
                     0429  1584 ; The message buffers and datagram buffers are inserted on the
                     0429  1585 ; port free queues for receiving messages and datagrams.
                     0429  1586 ;
                     0429  1587 ; Entry SCS$ALL_ALLBUF2 is called to allocate all except the SCS receive buffer.
                     0429  1588 ;
                     0429  1589 ; Inputs:
                     0429  1590 ;
                     0429  1591 ;       R3                      -Addr of CDT getting buffers
                     0429  1592 ;       R4                      -Addr of PDT
                     0429  1593 ;
                     0429  1594 ; Outputs:
                     0429  1595 ;
                     0429  1596 ;       R0                      -Status:  SS$_NORMAL, SS$INSFMEM
                     0429  1597 ;                                  (If insufficient pool, everything
                     0429  1598 ;                                   allocated so far is deallocated.)
                     0429  1599 ;       R1,R2                   -Destroyed
                     0429  1600 ;       other registers         -Preserved
                     0429  1601 ;-
                     0429  1602
                     0429  1603          .ENABL  LSB
                     0429  1604
                     0429  1605 SCS$ALL_ALLBUF::
                     0429  1606
          24   10    0429  1607          BSBB    SCS$ALL_SCSREC          ; Allocate SCS recv buffer
    1B 50   E9    042B  1608          BLBC    R0,40$                  ; Branch if error
                     042E  1609
                     042E  1610 SCS$ALL_ALLBUF2::
                     042E  1611
    48 A3   B0    042E  1612          MOVW    CDT$W_INITLREC(R3),-    ; Copy initial receive count to
    46 A3         0431  1613                  CDT$W_PENDREC(R3)       ;  pending credit to remote
    42 A3   B4    0433  1614          CLRW    CDT$W_REC(R3)           ; Clear current receive count
     0025   30    0436  1615          BSBW    SCS$ALL_MSGREC          ; Allocate message buffers
    0A 50   E9    0439  1616          BLBC    R0,30$                  ; Branch if error
     0071   30    043C  1617          BSBW    SCS$ALL_DGREC           ; Allocate datagram buffers
    01 50   E9    043F  1618          BLBC    R0,20$                  ; Branch if error
                  05    0442  1619          RSB                             ; Return success
                     0443  1620
     00B0   30    0443  1621 20$:     BSBW    SCS$DEAL_MSGREC         ; Deallocate message buffers
     00D4   30    0446  1622 30$:     BSBW    SCS$DEAL_SCSREC         ; Deallocate SCS recv buffer
                     0449  1623
50   0124 8F   3C    0449  1624 40$:     MOVZWL  #SS$_INSFMEM,R0         ; Set error status
                  05    044E  1625          RSB                             ; Return
                     044F  1626
                     044F  1627          .DSABL  LSB
```

```
                     044F  1629          .SBTTL  -       SCS$ALL_SCSREC, ALLOCATE A MESSAGE BUFFER FOR
                     044F  1630          .SBTTL  -                       HOLDING COPY OF SCS REQUESTS
                     044F  1631
                     044F  1632 ;+
                     044F  1633 ; This routine allocates one message buffer and links it to the
                     044F  1634 ; CDT$L_SCSMSG.
                     044F  1635 ;
                     044F  1636 ; Inputs:
                     044F  1637 ;
                     044F  1638 ;       R3                      -Addr of CDT getting buffer
                     044F  1639 ;
                     044F  1640 ; Outputs:
                     044F  1641 ;
                     044F  1642 ;       R0                      -Status:  SS$_NORMAL,  SS$_INSFMEM
                     044F  1643 ;       R1,R2                   -Destroyed
                     044F  1644 ;       other registers         -Preserved
                     044F  1645 ;-
                     044F  1646
                     044F  1647          .ENABL  LSB
                     044F  1648
                     044F  1649 SCS$ALL_SCSREC:
                     044F  1650
      FBAE'  30      044F  1651          BSBW    INT$ALLOC_MSG           ; Allocate 1 message buffer from pool
   04 50     E9      0452  1652          BLBC    R0,10$                  ; Branch if didn't get it
2C A3   52   D0      0455  1653          MOVL    R2,CDT$L_SCSMSG(R3)     ; Save its address in CDT
             05      0459  1654 10$:     RSB                             ; Return
                     045A  1655
                     045A  1656          .DSABL  LSB
```

```
                     045A  1658          .SBTTL  -       SCS$ALL_MSGREC, ALLOCATE BUFFERS FOR RECEIVING
                     045A  1659          .SBTTL  -                       APPLICATION MESSAGES
                     045A  1660          .SBTTL  -       SCS$ALL_FRMSGS, ALLOCATE MESSAGE BUFFERS AND
                     045A  1661          .SBTTL  -                       PUT ON PORT FREE QUEUE
                     045A  1662
                     045A  1663 ;+
                     045A  1664 ; This routine allocates the number of message buffers specified in
                     045A  1665 ; CDT$W_INITLREC or R0, linking them together as they are allocated.
                     045A  1666 ; If all are allocated successfully, then they are inserted on the
                     045A  1667 ; port free message queue and successful return is taken.
                     045A  1668 ; If there is an allocate failure, then the buffers allocated already
                     045A  1669 ; are returned to pool and error is returned to the caller.
                     045A  1670 ;
                     045A  1671 ; Inputs:
                     045A  1672 ;
                     045A  1673 ;       R0                      -# buffers to allocate (SCS$ALL_FRMSGS)
                     045A  1674 ;       R3                      -Addr of CDT (SCS$ALL_MSGREC entry)
                     045A  1675 ;       R4                      -Addr of PDT
                     045A  1676 ;
                     045A  1677 ; Outputs:
                     045A  1678 ;
                     045A  1679 ;       R0                      -Status:  SS$_NORMAL, SS$_INSFMEM
                     045A  1680 ;       R1,R2                   -Destroyed
                     045A  1681 ;       other registers         -Preserved
                     045A  1682 ;-
                     045A  1683
                     045A  1684          .ENABL  LSB
                     045A  1685
                     045A  1686 SCS$ALL_FRMSGS::
                     045A  1687
          50    D5   045A  1688          TSTL    R0                      ; Any buffers to allocate?
          04    11   045C  1689          BRB     5$                      ; Join common code
                     045E  1690
                     045E  1691 SCS$ALL_MSGREC:
                     045E  1692
    50    48 A3    3C   045E  1693          MOVZWL  CDT$W_INITLREC(R3),R0   ; Get # buffers to allocate
          2B   13   0462  1694 5$:     BEQL    40$                     ; Branch if 0
          55   DD   0464  1695          PUSHL   R5                      ; Save R5
          53   DD   0466  1696          PUSHL   R3                      ;  and R3
       53   50   D0   0468  1697          MOVL    R0,R3                   ; R3 will be buffer counter
          55   D4   046B  1698          CLRL    R5                      ; Set link to next buffer = 0
                     046D  1699
         FB90'  30   046D  1700 10$:     BSBW    INT$ALLOC_MSG           ; Allocate next buffer
        20 50   E9   0470  1701          BLBC    R0,50$                  ; Branch if failed
       62   55   D0   0473  1702          MOVL    R5,(R2)                 ; Link this buffer to last allocated
       55   52   D0   0476  1703          MOVL    R2,R5                   ; Set R5 to point to this buffer
         F1 53   F5   0479  1704          SOBGTR  R3,10$                  ; Branch if more to allocate
          53 8ED0   047C  1705          POPL    R3                      ; Restore R3
                     047F  1706
       52   55   D0   047F  1707 20$:     MOVL    R5,R2                   ; Get addr of next buffer
          08   13   0482  1708          BEQL    30$                     ; Branch if no more
       55   62   D0   0484  1709          MOVL    (R2),R5                 ; Update ptr to next buffer
         FB76'  30   0487  1710          BSBW    INT$INS_MFREEQ          ; Insert buffer on port free queue
          F3   11   048A  1711          BRB     20$                     ; Branch for next buffer
                     048C  1712
          55 8ED0   048C  1713 30$:     POPL    R5                      ; Restore saved register
                     048F  1714
```

```
                50   01    3C   048F  1715 40$:    MOVZWL  #SS$_NORMAL,R0          ; Set status to success
                           05   0492  1716          RSB                             ; Return
                                0493  1717
                52   55    D0   0493  1718 50$:    MOVL    R5,R2                   ; Get addr of next buffer
                     08    13   0496  1719          BEQL    NO_MSG_MEM              ; Branch if done
                55   62    D0   0498  1720          MOVL    (R2),R5                 ; Update ptr to next buffer
                   FB62'   30   049B  1721          BSBW    INI$DEAL_MSG            ; Return buffer to pool
                     F3    11   049E  1722          BRB     50$                     ; Check for more buffers
                                04A0  1723
                                04A0  1724 NO_MSG_MEM:
                                04A0  1725
              50   0124 8F    3C   04A0  1726          MOVZWL  #SS$_INSFMEM,R0         ; Set status to fail
                        53 8ED0  04A5  1727          POPL    R3                      ; Restore registers
                        55 8ED0  04A8  1728          POPL    R5                      ;
                           05   04AB  1729          RSB                             ; Return
                                04AC  1730
                                04AC  1731          .DSABL  LSB
```

```
                        04AC  1733          .SBTTL  -       SCS$ALL_DGREC,  ALLOCATE BUFFERS FOR RECEIVING
                        04AC  1734          .SBTTL  -                       APPLICATION DATAGRAMS
                        04AC  1735          .SBTTL  -       SCS$ALL_FRDGS,  ALLOCATE FREE DATAGRAMS AND
                        04AC  1736          .SBTTL  -                       PUT ON PORT QUEUE
                        04AC  1737
                        04AC  1738 ;+
                        04AC  1739 ; This routine allocates the number of datagrams specified in
                        04AC  1740 ; CDT$W_DGREC or R0, linking them together as allocated.  If all are allocated
                        04AC  1741 ; successfully, then they are inserted on the port free datagram queue.
                        04AC  1742 ; Otherwise, the buffers allocated already are returned to pool and error
                        04AC  1743 ; is returned to the caller.
                        04AC  1744 ;
                        04AC  1745 ; Inputs:
                        04AC  1746 ;
                        04AC  1747 ;       R0                      -# datagram buffers to allocate
                        04AC  1748 ;                                (SCS$ALL_FRDGS entry)
                        04AC  1749 ;       R3                      -Addr of CDT (SCS$ALL_DGREC entry)
                        04AC  1750 ;       R4                      -Addr of PDT
                        04AC  1751 ;
                        04AC  1752 ; Outputs:
                        04AC  1753 ;
                        04AC  1754 ;       R0                      -Status:  SS$_NORMAL, SS$INSFMEM
                        04AC  1755 ;       R1,R2                   -Destroyed
                        04AC  1756 ;       other registers         -Preserved
                        04AC  1757 ;-
                        04AC  1758
                        04AC  1759          .ENABL  LSB
                        04AC  1760
                        04AC  1761 SCS$ALL_FRDGS::
                        04AC  1762
                  50    D5  04AC  1763          TSTL    R0                      ; Any datagrams to get?
                  04    11  04AE  1764          BRB     5$                      ; Join common code
                        04B0  1765
                        04B0  1766 SCS$ALL_DGREC:
                        04B0  1767
            50    4C  A3    3C  04B0  1768          MOVZWL  CDT$W_DGREC(R3),R0      ; Get # buffers to allocate
                  2B    13  04B4  1769 5$:       BEQL    40$                     ; Branch if 0
                  55    DD  04B6  1770          PUSHL   R5                      ; Save R5
                  53    DD  04B8  1771          PUSHL   R3                      ;  and R3
               53     50    D0  04BA  1772          MOVL    R0,R3                   ; R3 will be buffer counter
                  55    D4  04BD  1773          CLRL    R5                      ; Set link to next buffer = 0
                        04BF  1774
                  FB3E'   30  04BF  1775 10$:      BSBW    INT$ALLOC_DG            ; Allocate next buffer
                20 50     E9  04C2  1776          BLBC    R0,50$                  ; Branch if failed
               62    55    D0  04C5  1777          MOVL    R5,(R2)                 ; Link this buffer to last allocated
               55    52    D0  04C8  1778          MOVL    R2,R5                   ; Set R5 to point to this buffer
                 F1 53     F5  04CB  1779          SOBGTR  R3,10$                  ; Branch if more to allocate
                  53 8ED0  04CE  1780          POPL    R3                      ; Retreive R3
                        04D1  1781
               52    55    D0  04D1  1782 20$:      MOVL    R5,R2                   ; Get addr of next buffer
                  08    13  04D4  1783          BEQL    30$                     ; Branch if no more
               55    62    D0  04D6  1784          MOVL    (R2),R5                 ; Update ptr to next buffer
                  FB24'   30  04D9  1785          BSBW    INT$INS_DFREEQ          ; Insert buffer on port free queue
                  F3    11  04DC  1786          BRB     20$                     ; Branch for next buffer
                        04DE  1787
                  55 8ED0  04DE  1788 30$:      POPL    R5                      ; Restore saved register
                        04E1  1789
```

```
                50   01    3C  04E1  1790 40$:    MOVZWL  #SS$_NORMAL,R0          ; Set status to success
                           05  04E4  1791         RSB                             ; Return
                               04E5  1792
                52   55    D0  04E5  1793 50$:    MOVL    R5,R2                   ; Get addr of next buffer
                     B6    13  04E8  1794         BEQL    NO_MSG_MEM              ; Branch if no more
                55   62    D0  04EA  1795         MOVL    (R2),R5                 ; Update ptr to next buffer
                   FB10'   30  04ED  1796         BSBW    INT$DEAL_DG             ; Return buffer to pool
                     F3    11  04F0  1797         BRB     50$                     ; Check for more buffers
                               04F2  1798
                               04F2  1799         .DSABL  LSB
```

```
                     04F2  1801          .SBTTL  -       SCS$DEAL_ALLBUF, DEALLOCATE ALL BUFFERS NEEDED
                     04F2  1802          .SBTTL  -                       FOR A CONNECTION
                     04F2  1803
                     04F2  1804 ;+
                     04F2  1805 ; DEALL_BUF deallocates the following to nonpaged pool:
                     04F2  1806 ;
                     04F2  1807 ;       1 SCS message buffer (linked to CDT$L_SCSMSG)
                     04F2  1808 ;       Initial receive credit worth of message buffers
                     04F2  1809 ;       DG receive count of datagram buffers
                     04F2  1810 ;
                     04F2  1811 ; It is the caller's responsibility to guarantee that all the message
                     04F2  1812 ; buffers and datagram buffers claimed to be on the port free queue
                     04F2  1813 ; are actually there.
                     04F2  1814 ;
                     04F2  1815 ; Inputs:
                     04F2  1816 ;
                     04F2  1817 ;       R3                      -Addr of CDT returning buffers
                     04F2  1818 ;       R4                      -Addr of PDT
                     04F2  1819 ;
                     04F2  1820 ; Outputs:
                     04F2  1821 ;
                     04F2  1822 ;       R0                      -Destroyed
                     04F2  1823 ;       other registers         -Preserved
                     04F2  1824 ;-
                     04F2  1825
                     04F2  1826          .ENABL  LSB
                     04F2  1827
                     04F2  1828 SCS$DEAL_ALLBUF::
                     04F2  1829
  29    10           04F2  1830          BSBB    SCS$DEAL_SCSREC         ; Deallocate the SCS buffer
                     04F4  1831                                          ;  attached to the CDT
  16    10           04F4  1832          BSBB    SCS$DEAL_DGREC          ; Deallocate the dg buffers
                     04F6  1833                                          ;  attached to the port free queue
                     04F6  1834 ;        BSBW    SCS$DEAL_MSGREC         ; Deallocate the message buffers
                     04F6  1835 ;                                        ;  attached to the port free queue
                     04F6  1836 ;        RSB                             ; Return
                     04F6  1837
                     04F6  1838          .DSABL  LSB
```

```
                     04F6  1840          .SBTTL  -       SCS$DEAL_MSGREC, DEALLOCATE BUFFERS QUEUED TO
                     04F6  1841          .SBTTL  -                       PORT FOR RECEIVING APPLICATION
                     04F6  1842          .SBTTL  -                       MESSAGES
                     04F6  1843
                     04F6  1844 ;+
                     04F6  1845 ; SCS$DEAL_MSGREC extracts the current recv credit worth of message buffers
                     04F6  1846 ; from the port free queue and returns them to pool.
                     04F6  1847 ;
                     04F6  1848 ; Inputs:
                     04F6  1849 ;
                     04F6  1850 ;       R3                      -Addr of CDT returning buffers
                     04F6  1851 ;       R4                      -Addr of PDT
                     04F6  1852 ;
                     04F6  1853 ; Outputs:
                     04F6  1854 ;
                     04F6  1855 ;       R0                      -Destroyed
                     04F6  1856 ;       other registers         -Preserved
                     04F6  1857 ;
                     04F6  1858 ;       CDT$W_REC(R3)           -0
                     04F6  1859 ;-
                     04F6  1860
                     04F6  1861          .ENABL  LSB
                     04F6  1862
                     04F6  1863 SCS$DEAL_MSGREC:
                     04F6  1864
   52    DD          04F6  1865          PUSHL   R2                      ; Save register
 46 A3   A0          04F8  1866          ADDW    CDT$W_PENDREC(R3),-     ; Compute total credit = pending
 42 A3               04FB  1867                  CDT$W_REC(R3)           ;  + receives given to remote
                     04FD  1868
 42 A3   B5          04FD  1869 10$:     TSTW    CDT$W_REC(R3)           ; Any buffers left to deallocate?
    29   13          0500  1870          BEQL    30$                     ; Branch if not
  FAFB'  30          0502  1871          BSBW    INT$MFQ2POOL            ; Else remove another from port
    24   1D          0505  1872          BVS     30$                     ; Branch if no more -- normally does't
                     0507  1873                                          ;  happen except during power fail recov
 42 A3   B7          0507  1874          DECW    CDT$W_REC(R3)           ; Decr count of # buffers left
    F1   11          050A  1875          BRB     10$                     ; Deallocate again
```

```
                     050C  1877          .SBTTL  -       SCS$DEAL_DGBUF, DEALLOCATE BUFFERS QUEUED TO PORT
                     050C  1878          .SBTTL  -                       FOR RECEIVING DATAGRAMS
                     050C  1879
                     050C  1880 ;+
                     050C  1881 ; SCS$DEAL_DGREC extracts the current number of datagrams contributed by
                     050C  1882 ; this connection to the free queue and returns them to nonpaged pool
                     050C  1883 ;
                     050C  1884 ; Inputs:
                     050C  1885 ;
                     050C  1886 ;       R3                      -Addr of CDT returning datagrams
                     050C  1887 ;       R4                      -Addr of PDT
                     050C  1888 ;
                     050C  1889 ; Outputs:
                     050C  1890 ;
                     050C  1891 ;       R0                      -Destroyed
                     050C  1892 ;       Other registers         -Preserved
                     050C  1893 ;-
                     050C  1894
                     050C  1895 SCS$DEAL_DGREC:
                     050C  1896
   52     DD         050C  1897          PUSHL   R2                      ; Save register
                     050E  1898
 4C A3    B5         050E  1899 20$:     TSTW    CDT$W_DGREC(R3)         ; Any buffers left to deallocate?
   18     13         0511  1900          BEQL    30$                     ; Branch if not
  FAEA'   30         0513  1901          BSBW    INT$DFQ2POOL            ; Remove another buffer to pool
   13     1D         0516  1902          BVS     30$                     ; Branch if no more-- normally doesn't
                     0518  1903                                          ;  happen except on power fail recovery
 4C A3    B7         0518  1904          DECW    CDT$W_DGREC(R3)         ; Decr count of # dgs contributed
                     051B  1905                                          ;  by this connection
   F1     11         051B  1906          BRB     20$                     ; Deallocate again
```

```
                     051D  1908          .SBTTL  -       SCS$DEAL_SCSREC, DEALLOCATE SCS RECEIVE BUFFER
                     051D  1909
                     051D  1910 ;+
                     051D  1911 ; SCS$DEAL_SCSREC deallocates to nonpaged pool the message buffer,
                     051D  1912 ; if any, pointed to by CDT$L_SCSMSG.
                     051D  1913 ;
                     051D  1914 ; Inputs:
                     051D  1915 ;
                     051D  1916 ;       R3                      -Addr of the CDT returning the buffer
                     051D  1917 ;
                     051D  1918 ; Outputs:
                     051D  1919 ;
                     051D  1920 ;       R0                      -Destroyed
                     051D  1921 ;       other registers         -Preserved
                     051D  1922 ;-
                     051D  1923
                     051D  1924 SCS$DEAL_SCSREC::
                     051D  1925
     52   DD         051D  1926          PUSHL   R2                              ; Save a register
52   2C A3   D0      051F  1927          MOVL    CDT$L_SCSMSG(R3),R2             ; Get buf'er address
        06   13      0523  1928          BEQL    30$                             ; Branch if none
      FAD8'  30      0525  1929          BSBW    INT$DEAL_MSG                    ; Return to pool
     2C A3   D4      0528  1930          CLRL    CDT$L_SCSMSG(R3)                ; Show no SCS buffer in CDT
                     052B  1931
        52 8ED0      052B  1932 30$:     POPL    R2                              ; Restore a register
             05      052E  1933          RSB                                     ; Return
                     052F  1934
                     052F  1935          .DSABL  LSB
```

```
                        .SBTTL  MISC ROUTINES
                        .SBTTL  -       BREAK_VC,       CRASH A VIRTUAL CIRCUIT

                ;+
                ; BREAK_VC is branched to from various SCS message receive routines
                ; when an SCS control message is received which is inappropriate
                ; to the current CDT state, has an illegal SCS message type, or performs
                ; unexpected action.  The remote system is assumed to have a broken SCS.
                ; The error is logged.  Routine ERR$CRASHVC is called to look up the path
                ; block associated with the message and close the virtual circuit to the
                ; remote.  The message in hand is occupying either: (1) the SCS send buffer
                ; in which virtual circuit failure expects to find it either in PB$L_SCSMSG
                ; or on the free queue, or (2) the SCS receive buffer in which case
                ; virtual circuit failure expects to find it on the free queue.  Action
                ; here is to return it to the free queue regardless of whether it is the
                ; SCS send or receive buffer.  Return is to interrupt service to dequeue
                ; the next response.
                ;
                ; Inputs:
                ;
                ;       R2                      -SCS message addr
                ;       R3                      -CDT addr
                ;       R4                      -PDT addr
                ;
                ; Outputs:
                ;
                ;       R0-R2                   -Destroyed
                ;       Other registers         -Perserved
                ;-

                        .ENABL  LSB

BREAK_VC:

51   1C A3      D0      MOVL    CDT$L_PB(R3),R1         ; Pick up Path Block address
   50   05      9A      MOVZBL  #PAER$K_ES_SCA, R0      ; Log inappropriate SCS - SCA control
      FAC7'     30      BSBW    ELOG$PACKET             ; message received error.
      FAC4'     30      BSBW    ERR$CRASHVC             ; Call error handler to init
                                                        ;  close of VC
      FAC1'     31      BRW     INT$INS_MFREEQ          ; Put msg back on free queue to
                                                        ;  be reclaimed when it is time
                                                        ;  to delete the path block
                                                        ; Return to interrupt service

                        .DSABL  LSB
```

```
                     053F  1983          .SBTTL  -       SCS$FREE_LISTEN,PUT BUSY LISTEN CDT
                     053F  1984          .SBTTL  -                       BACK IN LISTEN STATE
                     053F  1985
                     053F  1986 ;+
                     053F  1987 ; SCS$FREE_LISTEN deallocates the SCS receive buffer, if any, held
                     053F  1988 ; by the specified CDT, searches for the CDT in the path block
                     053F  1989 ; CDT list, removes the CDT from the list, and sets the CDT state
                     053F  1990 ; to LISTEN.
                     053F  1991 ;
                     053F  1992 ; Inputs:
                     053F  1993 ;
                     053F  1994 ;       R3                      -Addr of CDT
                     053F  1995 ;
                     053F  1996 ; Outputs:
                     053F  1997 ;
                     053F  1998 ;       R0                      -Destroyed
                     053F  1999 ;
                     053F  2000 ;       Other registers         -Preserved
                     053F  2001 ;
                     053F  2002 ;       CDT$W_STATE(R3)         -LISTEN state
                     053F  2003 ;-
                     053F  2004
                     053F  2005          .DSABL  LSB
                     053F  2006
                     053F  2007 SCS$FREE_LISTEN::
                     053F  2008
        51    DD     053F  2009          PUSHL   R1                      ; Save caller's R1
      FFD9    30     0541  2010          BSBW    SCS$DEAL_SCSREC         ; Deallocate SCS recv buffer, if any
51    1C A3   D0     0544  2011          MOVL    CDT$L_PB(R3),R1         ; Get PB associdated with CDT
51    34 A1   DE     0548  2012          MOVAL   PB$L_CDTLST(R1),R1      ; Get addr of 1st fwd link
                     054C  2013
   50   61    D0     054C  2014 10$:     MOVL    (R1),R0                 ; Get next CDT
        0F    13     054F  2015          BEQL    30$                     ; Branch if no more
   53   50    D1     0551  2016          CMPL    R0,R3                   ; CDT we are looking for?
        06    13     0554  2017          BEQL    20$                     ; Branch if so
51    6C A0   DE     0556  2018          MOVAL   CDT$L_CDTLST(R0),R1     ; Save addr of this link
        F0    11     055A  2019          BRB     10$                     ; Go for another CDT
                     055C  2020
61    6C A0   D0     055C  2021 20$:     MOVL    CDT$L_CDTLST(R0),(R1)   ; Unlink this CDT from list
                     0560  2022
        01    B0     0560  2023 30$:     MOVW    #CDT$C_LISTEN,-         ; Put CDT in listen state
      28 A3          0562  2024                  CDT$W_STATE(R3)         ;
        51 8ED0      0564  2025          POPL    R1                      ; Restore caller's R1
             05      0567  2026          RSB                             ; Return
                     0568  2027
                     0568  2028          .DSABL  LSB
```

```
                     0568  2030          .SBTTL  -       RESUME_CONCALL, RESUME A CONNECTION
                     0568  2031          .SBTTL  -                       MANAGEMENT CALL
                     0568  2032
                     0568  2033 ;+
                     0568  2034 ; RESUME_CONCALL resumes the fork process context saved in the CDT
                     0568  2035 ; during a CONNECT, ACCEPT, or REJECT call.
                     0568  2036 ;
                     0568  2037 ; Inputs:
                     0568  2038 ;
                     0568  2039 ;       R0                      -Completion status of connection
                     0568  2040 ;                                management call
                     0568  2041 ;       R1                      -Other status, if applicable
                     0568  2042 ;       R3                      -Addr of CDT
                     0568  2043 ;       R4                      -Addr of PDT
                     0568  2044 ;       (SP)                    -Return to PAINIT, interrupt service
                     0568  2045 ;
                     0568  2046 ; Outputs:
                     0568  2047 ;
                     0568  2048 ;       R0-R3,R5                -Destroyed
                     0568  2049 ;       Other registers         -Preserved
                     0568  2050 ;-
                     0568  2051
                     0568  2052          .ENABL  LSB
                     0568  2053
                     0568  2054 RESUME_CONCALL:
                     0568  2055
  55   68 A3    D0   0568  2056          MOVL    CDT$L_FR5(R3),R5        ; Restore SYSAP R5
       54       DD   056C  2057          PUSHL   R4                      ; Save R4
       64 B3    16   056E  2058          JSB     @CDT$L_FPC(R3)          ; Call SYSAP back
       54 8ED0  05   0571  2059          POPL    R4                      ; Restore R4
                05   0574  2060          RSB                             ; Return to interrupt service
                     0575  2061                                          ;  to dequeue more responses
                     0575  2062
                     0575  2063          .DSABL  LSB
```

```
                     0575  2065          .SBTTL  -       SCS$CHK_SEQNUM, CHECK FOR VALID
                     0575  2066          .SBTTL  -                       SEQUENCE # IN
                     0575  2067          .SBTTL  -                       CONNECTION ID
                     0575  2068
                     0575  2069 ;+
                     0575  2070 ; SCS$CHK_SEQNUM compares the sequence number fields in the
                     0575  2071 ; destination connection ID in a specifed SCS message and specified
                     0575  2072 ; CDT.  If they are equal, then the CDT is current for this message
                     0575  2073 ; and success is returned.  If they are not equal, then this CDT was
                     0575  2074 ; unilaterally closed sometime in the past and does not belong to
                     0575  2075 ; a current CDT and fail status is returned.
                     0575  2076 ;
                     0575  2077 ; Inputs:
                     0575  2078 ;
                     0575  2079 ;       R2                      -Addr of message/dg
                     0575  2080 ;       R3                      -Addr of CDT
                     0575  2081 ;
                     0575  2082 ; Outputs:
                     0575  2083 ;
                     0575  2084 ;       R0                      -Status:  LBS/C for CDT OK/invalid
                     0575  2085 ;       Other registers         -Preserved
                     0575  2086 ;-
                     0575  2087
                     0575  2088          .ENABL  LSB
                     0575  2089
                     0575  2090 SCS$CHK_SEQNUM:
                     0575  2091
   50    D4          0575  2092          CLRL    R0                      ; Assume failure
FA A2    B1          0577  2093          CMPW    SCS$L_DST_CONID+2(R2),- ; Compare seq # in dest conid
1A A3                057A  2094                  CDT$L_LCONID+2(R3)      ;  and in conid in CDT
   02    12          057C  2095          BNEQ    10$                     ; Branch if not the same (fail)
   50    D6          057E  2096          INCL    R0                      ; Else set status to success
                     0580  2097
         05          0580  2098 10$:     RSB                             ; Return status
                     0581  2099
                     0581  2100          .DSABL  LSB
```

```
                     0581  2102          .SBTTL  -       SCS$CLOSE_CDT,  CLEANUP CDT AND COMPLETE
                     0581  2103          .SBTTL  -                       PENDING CONNX MGMT CALL
                     0581  2104
                     0581  2105 ;+
                     0581  2106 ; SCS$CLOSE_CDT retreives the saved SYSAP context from the specified CDT
                     0581  2107 ; for a pending CONNECT/ACCEPT/REJECT/DISCONNECT call, deallocates the
                     0581  2108 ; CDT and msgs/dgs queued for receive, and completes the SYSAP's pending
                     0581  2109 ; call.
                     0581  2110 ;
                     0581  2111 ; Inputs:
                     0581  2112 ;
                     0581  2113 ;       R0                      -VMS status to complete pending call
                     0581  2114 ;       R1                      -Aux status, if any
                     0581  2115 ;       R3                      -CDT to clean up
                     0581  2116 ;       R4                      -Addr of PDT
                     0581  2117 ;
                     0581  2118 ; Outputs:
                     0581  2119 ;
                     0581  2120 ;       R0-R3                   -Destroyed
                     0581  2121 ;       Other registers         -Preserved
                     0581  2122 ;-
                     0581  2123
                     0581  2124          .ENABL  LSB
                     0581  2125
                     0581  2126 SCS$CLOSE_CDT::
                     0581  2127
      7E    54   7D  0581  2128          MOVQ    R4,-(SP)                ; Save R4, R5
55    68 A3   D0  0584  2129          MOVL    CDT$L_FR5(R3),R5        ; Get SYSAP context, R5
      64 A3   DD  0588  2130          PUSHL   CDT$L_FPC(R3)           ;  and PC
      7E    50   7D  058B  2131          MOVQ    R0,-(SP)                ; Save status
        FF61   30  058E  2132          BSBW    SCS$DEAL_ALLBUF         ; Deallocate receive buffers
00000000'GF   16  0591  2133          JSB     G^SCS$DEALL_CDT         ;  and CDT
      50    8E   7D  0597  2134          MOVQ    (SP)+,R0                ; Retreive status
            9E   16  059A  2135          JSB     @(SP)+                  ; Call SYSAP's return addr
      54    8E   7D  059C  2136          MOVQ    (SP)+,R4                ; Restore R4, R5
                 05  059F  2137          RSB                             ; Return
                     05A0  2138
                     05A0  2139          .DSABL  LSB
```

```
                     05A0  2141          .SBTTL  -       SCS$MAP_SCSSTS  MAP SCS STATUS TO VMS STATUS
                     05A0  2142
                     05A0  2143 ;+
                     05A0  2144 ; SCS$MAP_SCSSTS gets the SCS status contained in the SCS
                     05A0  2145 ; message pointed to by R2.  The corresponding VMS status is looked
                     05A0  2146 ; up and returned in R0.  (The status tables are contained in
                     05A0  2147 ; SCS_STATUS_TAB and VMS_STATUS_TAB.)  If the SCS status is not
                     05A0  2148 ; in the translation table, it is returned as is.
                     05A0  2149 ;
                     05A0  2150 ; Inputs:
                     05A0  2151 ;
                     05A0  2152 ;       R2                      -SCS message address
                     05A0  2153 ;
                     05A0  2154 ; Outputs:
                     05A0  2155 ;
                     05A0  2156 ;       R0                      -L.o. word:  converted status
                     05A0  2157 ;                                or SYSAP-specific status.  H.o.
                     05A0  2158 ;                                word = 0.
                     05A0  2159 ;       Other registers         -Preserved
                     05A0  2160 ;-
                     05A0  2161
                     05A0  2162          .ENABL  LSB
                     05A0  2163
                     05A0  2164 SCS$MAP_SCSSTS:
                     05A0  2165
        0E   BB      05A0  2166          PUSHR   #^M<R1,R2,R3>           ; Save registers
02 A2   02   39      05A2  2167          MATCHC  #2,SCS$W_STATUS(R2),-   ; Look up status in
        12           05A6  2168                  #SCS_STATUS_LEN,-       ;  SCS status list
   FA56 CF           05A7  2169                  SCS_STATUS_TAB          ;
        07   12      05AA  2170          BNEQ    20$                     ; Branch if did not find it
50   10 A3   3C      05AC  2171          MOVZWL  SCS_STATUS_LEN-2(R3),R0 ; Get corresponding VMS status
        0E   BA      05B0  2172          POPR    #^M<R1,R2,R3>           ; Restore registers
             05      05B2  2173          RSB                             ; Return
                     05B3  2174
        0E   BA      05B3  2175 20$:     POPR    #^M<R1,R2,R3>           ; Restore registers
50   02 A2   3C      05B5  2176          MOVZWL  SCS$W_STATUS(R2),R0     ; Return original status
             05      05B9  2177          RSB                             ; Return
                     05BA  2178
                     05BA  2179          .DSABL  LSB
```

```
                                05BA  2181          .SBTTL  -       SCS$MAP_VMSSTS, MAP VMS STATUS TO SCS
                                05BA  2182
                                05BA  2183 ;+
                                05BA  2184 ; SCS$MAP_VMSSTS converts a VMS status code to the corresponding SCS
                                05BA  2185 ; code.  If the VMS code is not in the conversion table, then the VMS
                                05BA  2186 ; code is returned unchanged.
                                05BA  2187 ;
                                05BA  2188 ; Inputs:
                                05BA  2189 ;
                                05BA  2190 ;       R0                          -VMS code (l.o. 16 bits only)
                                05BA  2191 ;
                                05BA  2192 ; Outputs:
                                05BA  2193 ;
                                05BA  2194 ;       R0                          -SCS code, if conversion is successful,
                                05BA  2195 ;                                    otherwise, VMS code
                                05BA  2196 ;       Other registers             -Preserved
                                05BA  2197 ;-
                                05BA  2198          ;
                                05BA  2199          .ENABL  LSB
                                05BA  2200
                                05BA  2201 SCS$MAP_VMSSTS::
                                05BA  2202
                      0F    BB  05BA  2203          PUSHR   #^M<R0,R1,R2,R3>        ; Save caller's registers
                12    6E    02  39  05BC  2204          MATCHC  #2,(SP),#SCS_STATUS_LEN,-  ; Look up R0 status in table
                    FA4F    CF      05C0  2205                  VMS_STATUS_TAB          ;
                            09  12  05C3  2206          BNEQ    20$                     ; Branch if didn't find it
                 50   EC    A3  3C  05C5  2207          MOVZWL  -SCS_STATUS_LEN-2(R3),R0 ;Get corresponding SCS code
                            8E  D5  05C9  2208          TSTL    (SP)+                   ; Discard caller's R0
                            0E  BA  05CB  2209          POPR    #^M<R1,R2,R3>           ; Restore other registers
                                05  05CD  2210          RSB                             ; Return
                                    05CE  2211
                            0F  BA  05CE  2212 20$:     POPR    #^M<R0,R1,R2,R3>        ; Restore all caller's registers
                                05  05D0  2213          RSB                             ; Return
                                    05D1  2214
                                    05D1  2215          .DSABL  LSB
```

```
                     05D1  2217          .SBTTL  ERROR HANDLING ROUTINES
                     05D1  2218          .SBTTL  -       SCS$DISC_VCFAIL, PROCESS DISCONNECT CALL
                     05D1  2219          .SBTTL  -                        FOR CDT ON FAILING PB
                     05D1  2220
                     05D1  2221 ;+
                     05D1  2222 ; SCS$DISC_VCFAIL is called by FPC$DCONNECT when the SYSAP issues a
                     05D1  2223 ; DISCONNECT for a connection associated with a path block that has
                     05D1  2224 ; a virtual circuit failure in progress.  The CDT is placed in a
                     05D1  2225 ; virtual circuit fail state.  If this is the last CDT on the path
                     05D1  2226 ; block to be DISCONNECTed, then a marker message is sent to the
                     05D1  2227 ; port.  Receipt of the error response for the marker (handled in
                     05D1  2228 ; SCS$CACHECLR) tells us that the port cache is completely purged
                     05D1  2229 ; of all command queue entries for this virtual circuit.
                     05D1  2230 ;
                     05D1  2231 ; Inputs:
                     05D1  2232 ;
                     05D1  2233 ;       IPL                     -Fork IPL
                     05D1  2234 ;
                     05D1  2235 ;       R1                      -Addr of PB
                     05D1  2236 ;       R3                      -Addr of CDT being DISCONNECTed
                     05D1  2237 ;       R4                      -Addr of PDT
                     05D1  2238 ;
                     05D1  2239 ;       CDT$W_STATE(R3)         -Any except CLOSED
                     05D1  2240 ;
                     05D1  2241 ;       (SP)                    -Addr of return to SYSAP
                     05D1  2242 ;
                     05D1  2243 ; Outputs:
                     05D1  2244 ;
                     05D1  2245 ;       R0-R2                   -Destroyed
                     05D1  2246 ;       Other registers         -Preserved
                     05D1  2247 ;-
                     05D1  2248
                     05D1  2249          .ENABL  LSB
                     05D1  2250
                     05D1  2251 SCS$DISC_VCFAIL::
                     05D1  2252
                     05D1  2253          $DISPATCH       -                   ; Dispathc on CDT state:
                     05D1  2254                  CDT$W_STATE(R3),-           ;
                     05D1  2255                  <-                          ;
                     05D1  2256                  <CDT$C_CON_REC, 20$>,-      ;  Connect received on LISTEN,
                     05D1  2257                  <CDT$C_REJ_SENT,20$>,-      ;  Connect received on LISTEN,
                     05D1  2258                  <CDT$C_VC_FAIL, 30$>,-      ;  DISCONNECT already isssued,
                     05D1  2259                  >                           ;  All other states:
                     05DE  2260
          0C   B0    05DE  2261          MOVW    #CDT$C_VC_FAIL,-            ; Set state to VC failure
    28 A3            05E0  2262                  CDT$W_STATE(R3)             ;
                     05E2  2263
        00F4   30    05E2  2264 10$:     BSBW    CHK_NO_CDTS                 ; Send cache clear if all
                     05E5  2265                                              ;  CDTs have been disconnected
68 A3    55   D0    05E5  2266          MOVL    R5,CDT$L_FR5(R3)            ; Suspend this DISCONNECT
    64 A3 8ED0   D0  05E9  2267          POPL    CDT$L_FPC(R3)               ;  until cache is clear
              05     05ED  2268          RSB                                 ; Return to caller's caller
                     05EE  2269
        FF4E   30    05EE  2270 20$:     BSBW    SCS$FREE_LISTEN             ; Put CDT back in LISTEN state
          EF   11    05F1  2271          BRB     10$                         ; Join common check for no CDT's
                     05F3  2272
    50   01   3C    05F3  2273 30$:     MOVZWL  #SS$_NORMAL,R0              ; Return success to SYSAP
```

```
                -    FOR CDT ON FAILING PB

 05  05F6  2274          RSB                             ;
     05F7  2275
     05F7  2276          .DSABL  LSB
```

```
                     2278          .SBTTL  -       SCS$VC_CLOSED,  HANDLE VC CLOSED ON
                     2279          .SBTTL  -                       SPECIFIED PATH BLOCK
                     2280
                     2281 ;+
                     2282 ; SCS$VC_CLOSED checks if a vc failure is already underway.  This could
                     2283 ; happen if a REQID immediately followed by a SNDMSG both discover
                     2284 ; that there is no path to the target.  The REQID initiates vc
                     2285 ; closure by sending a SETCKT closed command to the port.  The
                     2286 ; SNDMSG then tries to do a cache clear while the SETCKT is still
                     2287 ; in progress.  The SETCKT and the cache clear need to use the
                     2288 ; same packet.  Further, if this routine were called twice in this
                     2289 ; situation, then sysap's would be notified twice.  Therefore, if
                     2290 ; a vc failure is underway, simply return, letting the first vc
                     2291 ; circuit take care of all the bookkeeping.
                     2292 ;
                     2293 ; SCS$VCCLOSED checks if there are any CDT's linked on the PB.
                     2294 ; If not, it calls CNF$REMOVE_PB to clean up the configuration.
                     2295 ; If so, it calls SCS$NOTIFY_SYSAP to notify all SYSAPs involved.
                     2296 ;
                     2297 ; SCS$SETCKT_CLSD is a separate entry to this routine which bypasses
                     2298 ; the check if a vc failure is in progress.  SCS$SETCKT_CLSD is called
                     2299 ; by PAINTR upon receipt of a SETCKT closed response.  A vc failure should
                     2300 ; always be marked in progress  in this case.
                     2301 ;
                     2302 ; Inputs:
                     2303 ;
                     2304 ;       IPL                     -Fork IPL
                     2305 ;
                     2306 ;       R1                      -Addr of failing PB, 0 if none
                     2307 ;                                (on SCS$SETCKT_CLSD there will always be
                     2308 ;                                a PB.)
                     2309 ;       R2                      -Addr of RSP
                     2310 ;       R3                      -Failure reason (VMS status code)
                     2311 ;       R4                      -PDT addr
                     2312 ;
                     2313 ;       VC state                -Closed by port
                     2314 ;
                     2315 ; Outputs:
                     2316 ;
                     2317 ;       R0,R1,R3                -Destroyed
                     2318 ;       Other registers         -Preserved;  in particular, the msg/dg
                     2319 ;                                pointed to is not disposed of.  That is
                     2320 ;                                the caller's responsibility.
                     2321 ;-
                     2322
                     2323          .ENABL  LSB
                     2324
                     2325 SCS$VCCLOSED::
                     2326
     51    D5        2327          TSTL    R1                      ; Check PB addr for remote port
     2F    13        2329          BEQL    CONFIG_ERR              ; There is none, log error
   12 A1   B1        2329          CMPW    PB$W_STATE(R1),-        ; Is a vc failure already in
 8000 8F             2330                  #PB$C_VC_FAIL           ;  progress?
     26    13        2331          BEQL    40$                     ; Branch if so.
                     2332
                     2333 SCS$SETCKT_CLSD::
                     2334
```

```
                52   DD  0603  2335          PUSHL   R2                              ; Save caller's register
          8000  8F   B0  0605  2336          MOVW    #PB$C_VC_FAIL,-                 ; Set VC failure in progess
            12  A1        0609  2337                  PB$W_STATE(R1)                  ;
            34  A1   D5  060B  2338          TSTL    PB$L_CDTLST(R1)                 ; Any CDT's on this PB?
                08   12  060E  2339          BNEQ    20$                             ; Branch if so
          53    51   D0  0610  2340          MOVL    R1,R3                           ; Transfer PB address
             F9EA'  30   0613  2341          BSBW    CNF$REMOVE_PB                   ; Else remove PB/SB
                0E   11  0616  2342          BRB     30$                             ; Return
                        0618  2343
       50    46  A1   3C  0618  2344 20$:    MOVZWL  PB$W_VCFAIL_RSN(R1),R0          ; Get possible remote host shutdown
                        061C  2345                                                  ;  reason for vc failure
                03   12  061C  2346          BNEQ    25$                             ; Branch if there is a reason saved
          50    53   D0  061E  2347          MOVL    R3,R0                           ; Else just copy the status handed
                        0621  2348                                                  ;  as input to this routine
                        0621  2349
          53    51   D0  0621  2350 25$:    MOVL    R1,R3                           ; Transfer path block
                22   10  0624  2351          BSBB    SCS$NOTIFY_SYSAP                ; Handle all CDT's in list
                52 8ED0  0626  2352 30$:    POPL    R2                              ; Restore caller's register
                        0629  2353
                     05  0629  2354 40$:    RSB                                     ; Return to caller to drain
                        062A  2355                                                  ;  response
                        062A  2356
                        062A  2357 CONFIG_ERR:
                        062A  2358
                        062A  2359          $DEBUGCHECK #ERR$V_DEB_CNFER            ; Optionally bugcheck on this error
       50   8006  8F   32  063D  2360          CVTWL   #<PAER$K_ES_NOPB ! ^X8000>, R0  ; Log failure to find PB for
             F9BB'  30   0642  2361          BSBW    ELOG$PACRET                     ;  given message while closing
                        0645  2362                                                  ;  VC error (crashes port).
             F9B8'  31   0645  2363          BRW     ERR$CRASH_PORT                  ; Init port crash
                        0648  2364
                        0648  2365          .DSABL  LSB
```

```
0648  2367          .SBTTL  -       SCS$NOTIFY_SYSAP, SEARCH CDT LIST AND
0648  2368          .SBTTL  -                        HANDLE CDT'S IN
0648  2369          .SBTTL  -                        VARIOUS STATES
0648  2370
0648  2371 ;+
0648  2372 ; NOTIFY_SYSAP is called by SCS$VCCLOSED and ERR$PWF_RECOV to search the
0648  2373 ; CDT list associated with a path block and notify the SYSAP appropriately
0648  2374 ; depending upon the CDT state.  It scans the CDT list for that path.  For
0648  2375 ; each open CDT found, the SYSAP's error entry is called to notify the
0648  2376 ; SYSAP of the virtual circuit failure.  The SYSAP may, in the course
0648  2377 ; of reclaiming threads (CDRP's) in progress, cause new traffic to be
0648  2378 ; sent to the port.  This happens with apparent success, e.g., new
0648  2379 ; SEND_MSG_BUF's do not yield errors.  The SYSAP may issue a DISCONNECT
0648  2380 ; as part of its error routine, or may defer the DISCONNECT till later.
0648  2381 ;
0648  2382 ; When all SYSAP's owning open CDT's have had their error entries called,
0648  2383 ; then SCS$VCCLOSED calls CDF$REMOVE_PB to claen up the configuration
0648  2384 ; database if there were no CDT's on this path block.
0648  2385 ;
0648  2386 ; CDT's in non-open states are handled in various ways:
0648  2387 ;
0648  2388 ;       State                   Action
0648  2389 ;
0648  2390 ;       CLOSED                  Nonfatal bugcheck since a closed CDT
0648  2391 ;                               should never be linked to a PB.
0648  2392 ;
0648  2393 ;       CON_ACK                 Terminate the connect call with
0648  2394 ;                               VCBROKEN status
0648  2395 ;
0648  2396 ;       DISC_ACK                Terminate disconnect with success status
0648  2397 ;
0648  2398 ;       CON_REC                 Call the SYSAP's error routine.
0648  2399 ;
0648  2400 ;       DISC_REC                Ignore this CDT since the SYSAP will
0648  2401 ;                               eventually DISCONNECT anyway.
0648  2402 ;
0648  2403 ;       CON_SENT                Terminate connect call with VCBROKEN
0648  2404 ;                               status
0648  2405 ;
0648  2406 ;       DISC_SENT               Terminate disconnect with success status
0648  2407 ;
0648  2408 ;       REJ_SENT                Terminate reject with success status and
0648  2409 ;                               return CDT to listening state.
0648  2410 ;
0648  2411 ;       ACCP_SENT               Terminate accept call with VCBROKEN status
0648  2412 ;
0648  2413 ;       LISTEN                  Nonfatal bugcheck since a listen CDT
0648  2414 ;                               should never be linked to the PB
0648  2415 ;
0648  2416 ;       DISC_MTCH               Terminate disconnect with success status
0648  2417 ;
0648  2418 ; Inputs:
0648  2419 ;
0648  2420 ;       R0                      -Aux status to pass to SYSAP
0648  2421 ;       R3                      -PB addr
0648  2422 ;       R4                      -PDT addr
0648  2423 ;
```

```
                     0648  2424 ; Outputs:
                     0648  2425 ;
                     0648  2426 ;       R0-R3                   -Destroyed
                     0648  2427 ;       Other registers         -Preserved
                     0648  2428 ;-
                     0648  2429
                     0648  2430         .ENABL  LSB
                     0648  2431
                     0648  2432 SCS$NOTIFY_SYSAP::
                     0648  2433
          55   DD    0648  2434         PUSHL   R5                      ; Save caller's R5
     51   50   D0    064A  2435         MOVL    R0,R1                   ; Transfer aux status
50   219C 8F   3C    064D  2436         MOVZWL  #SS$_VCBROKEN,R0        ; Set status to report to SYSAP
     55   53   D0    0652  2437         MOVL    R3,R5                   ; Save PB addr
  53   34 A3   D0    0655  2438         MOVL    PB$L_CDTLST(R3),R3      ; Get 1st CDT
          7A   13    0659  2439         BEQL    30$                     ; Branch if none
                     065B  2440
                     065B  2441 NEXT_CDT:
                     065B  2442
          6C A3   DD 065B  2443         PUSHL   CDT$L_CDTLST(R3)        ; Save addr of next CDT in case
                     065E  2444                                         ;  this one is deleted
                     065E  2445         $DISPATCH       -               ; Dispatch on CDT state
                     065E  2446                 CDT$W_STATE(R3),-       ;
                     065E  2447                 <-                      ;
                     065E  2448                 <CDT$C_OPEN,    CALL_ERRADDR>,- ;
                     065E  2449                 <CDT$C_CON_ACK, CONNECT_ABO>,-  ;
                     065E  2450                 <CDT$C_DISC_ACK,DCONNECT_OK>,-  ;
                     065E  2451                 <CDT$C_CON_REC, CALL_ERRADDR>,- ;
                     065E  2452                 <CDT$C_DISC_REC,IGNORE_CDT>,-   ;
                     065E  2453                 <CDT$C_CON_SENT,CONNECT_ABO>,-  ;
                     065E  2454                 <CDT$C_DISC_SENT,DCONNECT_OK>,- ;
                     065E  2455                 <CDT$C_REJ_SENT,REJECT_OK>,-    ;
                     065E  2456                 <CDT$C_ACCP_SENT,CONNECT_ABO>,- ;
                     065E  2457                 <CDT$C_DISC_MTCH,DCONNECT_OK>,- ;
                     065E  2458                 <CDT$C_VC_FAIL, IGNORE_CDT>,-   ;
                     065E  2459                 >
                     0679  2460
                     0679  2461         BUGCHECK        CIPORT,NONFATAL ; Illegal CDT state
                     0680  2462
  50   C8 A5   DE    0680  2463         MOVAL   PB$L_CDTLST-CDT$L_CDTLST(R5),R0
                     0684  2464                                         ; If nonfatal bugcheck, get PB
                     0684  2465                                         ;  CDT listhead addr - CDT$L_CDTLST
                     0684  2466
                     0684  2467 FIND_PRV_CDT:
                     0684  2468
  53   6C A0   D1    0684  2469         CMPL    CDT$L_CDTLST(R0),R3     ; Got previous link?
          06   13    0688  2470         BEQL    UNLNK_CDT               ; Branch if so
  50   6C A0   D0    068A  2471         MOVL    CDT$L_CDTLST(R0),R0     ; Else get next CDT
          F4   11    068E  2472         BRB     FIND_PRV_CDT            ; Go check it
                     0690  2473
                     0690  2474 UNLNK_CDT:
                     0690  2475
  6C A0   6E   D0    0690  2476         MOVL    (SP),CDT$L_CDTLST(R0)   ; Remove this CDT from list and
                     0694  2477                                         ;  take no further action on it
50   219C 8F   3C    0694  2478         MOVZWL  #SS$_VCBROKEN,R0        ; Retreive status
          35   11    0699  2479         BRB     GET_NEXT_CDT            ; Go for next CDT
                     069B  2480
```

```
                     069B  2481 CALL_ERRADDR:                          ; Notify SYSAP via error routine:
                     069B  2482
            33   BB  069B  2483          PUSHR   #^M<R0,R1,R4,R5>     ; Save registers we need
         0C B3   16  069D  2484          JSB     @CDT$L_ERRADDR(R3)   ; Call SYSAP error routine
            33   BA  06A0  2485          POPR    #^M<R0,R1,R4,R5>     ; Restore registers
            2C   11  06A2  2486          BRB     GET_NEXT_CDT         ; Join common code
                     06A4  2487
                     06A4  2488 CONNECT_ABO:                          ; Terminate pending connect/accept
                     06A4  2489                                       ;  with error status
                     06A4  2490
      7E     50   7D  06A4  2491          MOVQ    R0,-(SP)             ; Save status values
            1C   11  06A7  2492          BRB     10$                  ; Join common code
                     06A9  2493
                     06A9  2494 REJECT_OK:                            ; Terminate pending connection mgmt
                     06A9  2495                                       ;  call with success status, return
                     06A9  2496                                       ;  CDT to listening state.
      7E     50   7D  06A9  2497          MOVQ    R0,-(SP)             ; Save R0, R1
      50     01   3C  06AC  2498          MOVZWL  #SS$_NORMAL,R0       ; Set R0 to success status
            3C   BB  06AF  2499          PUSHR   #^M<R2,R3,R4,R5>     ; Save remaining registers
   55    68 A3   D0  06B1  2500          MOVL    CDT$L_FR5(R3),R5     ; Get sysap's R5
         64 B3   16  06B5  2501          JSB     @CDT$L_FPC(R3)       ; Resume sysap after its reject call
            3C   BA  06B8  2502          POPR    #^M<R2,R3,R4,R5>     ; Restore registers
           FE82  30  06BA  2503          BSBW    SCS$FREE_LISTEN      ; Put CDT back in listening state
            09   11  06BD  2504          BRB     20$                  ; Join common cleanup for next sysap
                     06BF  2505                                       ;  to notify
                     06BF  2506
                     06BF  2507 DCONNECT_OK:                          ; Terminate pending connection mgmt
                     06BF  2508                                       ;  call with success status:
                     06BF  2509
      7E     50   7D  06BF  2510          MOVQ    R0,-(SP)             ; Save standard status values
      50     01   3C  06C2  2511          MOVZWL  #SS$_NORMAL,R0       ; Set status to success
           FEB9   30  06C5  2512 10$:     BSBW    SCS$CLOSE_CDT        ; Complete pending disconnect call
                     06C8  2513
      51     55   D0  06C8  2514 20$:     MOVL    R5,R1                ; Retreive PB addr
            0C   10  06CB  2515          BSBB    CHK_NO_CDTS          ; Check CDTs disconnected now
                     06CD  2516                                       ;  and send cache clear if all disconnected
      50     8E   7D  06CD  2517          MOVQ    (SP)+,R0             ; Restore standard status values
                     06D0  2518
                     06D0  2519 IGNORE_CDT:
                     06D0  2520 GET_NEXT_CDT:
                     06D0  2521
            53 8ED0  06D0  2522          POPL    R3                   ; Retreive addr of next CDT
            86   12  06D3  2523          BNEQ    NEXT_CDT             ; Else go process
                     06D5  2524
            55 8ED0  06D5  2525 30$:     POPL    R5                   ; Restore caller's R5
                 05  06D8  2526          RSB                          ; Return
                     06D9  2527
                     06D9  2528          .DSABL  LSB
```

```
                     06D9  2530          .SBTTL  -       CHK_NO_CDTS,     IF ALL CDTS ON PB DISCONNECTED,
                     06D9  2531          .SBTTL  -                       SEND CACHE CLEAR MSG
                     06D9  2532
                     06D9  2533 ;+
                     06D9  2534 ; This routine is called by SCS$NOTIFY_SYSAP as partially open CDTs
                     06D9  2535 ; are closed out during a VC failure.  It checks whether any CDTs
                     06D9  2536 ; remain which have not been DISCONNECTed.  If so, CHK_NO_CDTS returns.
                     06D9  2537 ; If all have been disconnected, then check if the port is alive and
                     06D9  2538 ; able to process commands.  If the port is alive, then queue a cache
                     06D9  2539 ; clear message to it -- when the cache clear message has made it
                     06D9  2540 ; through the port, then we know we have retreive everything from the
                     06D9  2541 ; port's caches.
                     06D9  2542 ;
                     06D9  2543 ; If the port is dead (it was crashed or power failed), then a cache
                     06D9  2544 ; clear will not be processed.  In this case proceed as if the cache
                     06D9  2545 ; is already cleared (may lose some buffers this way.)  If there are
                     06D9  2546 ; still connections on this PB, return since future loops through
                     06D9  2547 ; NOTIFY_SYSAP or DISCONNECT will complete cleanup.  If all the
                     06D9  2548 ; connections are gone on this PB, then clear port queues of any
                     06D9  2549 ; commands or responses again and remove the path block from the
                     06D9  2550 ; configuration.  When there are no more path blocks
                     06D9  2551 ; (virtual circuits) associated with this port, then attempt a reinit
                     06D9  2552 ; ofthe port.  The port reinit may not be possible at this time due
                     06D9  2553 ; to port power loss and no restoration -- in this case, the reinit
                     06D9  2554 ; will happen on power up interrupt from the port.
                     06D9  2555 ;
                     06D9  2556 ; Inputs:
                     06D9  2557 ;
                     06D9  2558 ;       R1                      -Addr of PB
                     06D9  2559 ;       R4                      -Addr of PDT
                     06D9  2560 ;
                     06D9  2561 ; Outputs:
                     06D9  2562 ;
                     06D9  2563 ;       R0-R2                   -Destroyed
                     06D9  2564 ;       Other registers         -Preserved
                     06D9  2565 ;
                     06D9  2566 ;-
                     06D9  2567
                     06D9  2568          .ENABL  LSB
                     06D9  2569
                     06D9  2570 CHK_NO_CDTS:
                     06D9  2571
50   C8 A1   DE     06D9  2572          MOVAL   PB$L_CDTLST-CDT$L_CDTLST(R1),R0
                     06DD  2573                                          ; Get addr of CDT list - offset
                     06DD  2574                                          ;  CDT$L_CDTLST
                     06DD  2575
50   6C A0   D0     06DD  2576 10$:     MOVL    CDT$L_CDTLST(R0),R0     ; Get next CDT
        07   13     06E1  2577          BEQL    20$                     ; Branch if none
     28 A0   B1     06E3  2578          CMPW    CDT$W_STATE(R0),-       ; Is CDT state other than
        0C          06E6  2579                  #CDT$C_VC_FAIL          ;  VC fail?
        F4   13     06E7  2580          BEQL    10$                     ; Branch if not
             05     06E9  2581          RSB                             ; Else return without doing anything
                    06EA  2582                                          ;  since not all CDTs disconnected yet
                    06EA  2583
        00   E0     06EA  2584 20$:     BBS     #PDT$V_PWF_CLNUP,-      ; Is port processing commands or
03 0110 C4          06EC  2585                  PDT$W_PORT_STS(R4),30$  ;  crashed?  Branch if crashed.
      F90D'  31     06F0  2586          BRW     INT$CRCACHE             ; Else call PPD layer to clear out caches
```

```
                 06F3
      53   DD    06F3    30$:   PUSHL   R3                      ; Save caller's R3
   34 A1   D5    06F5           TSTL    PB$L_CDTLST(R1)         ; Any CDT's left on this PB?
      14   12    06F8           BNEQ    40$                     ; Branch if so
      51   DD    06FA           PUSHL   R1                      ; Save PB address
   F901'   30    06FC           BSBW    ERR$CLEANUP_PKT         ; Clear port queues one more time
                 06FF                                           ;  just in case
      53 8ED0    06FF           POPL    R3                      ; Get PB addr in right register
   F8FB'   30    0702           BSBW    CNF$REMOVE_PB           ; Remove path block from database
 0112 C4   B5    0705           TSTW    PDT$W_PBCOUNT(R4)       ; Any PB's on this port left?
      03   12    0709           BNEQ    40$                     ; Branch if so
   F8F2'   30    070B           BSBW    ERR$INIPORT             ; Else go call reinit of port if
                 070E                                           ;  port has power now
                 070E
      53 8ED0    070E    40$:   POPL    R3                      ; Restore caller's R3
         05      0711           RSB                             ; Return
                 0712
                 0712           .DSABL  LSB
```

```
                     0712  2606          .SBTTL  -       SCS$CACHECLR,   PROCESS REC'D MARKER MSG THAT
                     0712  2607          .SBTTL  -                       SAYS THE PORT CACHE IS CLEAR
                     0712  2608          .SBTTL  -                       OF TRAFFIC ON SPECIFIED VC
                     0712  2609
                     0712  2610 ;+
                     0712  2611 ; SCS$CACHECLR is called by PAINTR upon completion of any cache clearing
                     0712  2612 ; operations that were begun by CHK_NO_CDTS above.
                     0712  2613 ;
                     0712  2614 ; Inputs:
                     0712  2615 ;
                     0712  2616 ;       IPL                     -Fork IPL
                     0712  2617 ;
                     0712  2618 ;       R1                      -Addr of PB
                     0712  2619 ;       R4                      -Addr of PDT
                     0712  2620 ;
                     0712  2621 ; Outputs:
                     0712  2622 ;
                     0712  2623 ;       R0-R3                   -Destroyed
                     0712  2624 ;       Other registers         -Preserved
                     0712  2625 ;-
                     0712  2626
                     0712  2627          .ENABL  LSB
                     0712  2628
                     0712  2629 SCS$CACHECLR::
                     0712  2630
           51   D5  0712  2631          TSTL    R1                      ; Check PB  assoc with remote
                     0714  2632                                          ;  port in CACHECLR msg
           03   12  0714  2633          BNEQ    10$                     ; Branch if lookup successful
         FF11   31  0716  2634          BRW     CONFIG_ERR              ; Else, config database is
                     0719  2635                                          ;  inconsistent
                     0719  2636
              51   DD  0719  2637 10$:     PUSHL   R1                      ; Save PB addr
           53 51   D0  071B  2638          MOVL    R1,R3                   ; Retreive PB addr
        53   34 A3   D0  071E  2639          MOVL    PB$L_CDTLST(R3),R3      ; Get addr of 1st CDT
              0E   13  0722  2640          BEQL    50$                     ; Branch if none
                     0724  2641
           6C A3   DD  0724  2642 20$:     PUSHL   CDT$L_CDTLST(R3)        ; Save addr of next CDT
        50   01   3C  0727  2643          MOVZWL  #SS$_NORMAL,R0          ; Return success to suspended
             FE54   30  072A  2644          BSBW    SCS$CLOSE_CDT           ;  SYSAP and delete CDT
              53 8ED0  072D  2645          POPL    R3                      ; Get next CDT addr
              F2   12  0730  2646          BNEQ    20$                     ; Branch if got one
                     0732  2647
              53 8ED0  0732  2648 50$:     POPL    R3                      ; Retreive the PB addr
           F8C8'  31  0735  2649          BRW     CNF$REMOVE_PB           ; Deallocate PB/SB and return
                     0738  2650
                     0738  2651          .DSABL  LSB
                     0738  2652
                     0738  2653
                     0738  2654
                     0738  2655
                     0738  2656
                     0738  2657          .END
```

```
$$$CURSIZ                       =  000001C4
$$$NEWSIZ                       =  000001D0
ACCP_ABORT                         000001FB  R      01
BAD_DATABASE                       0000013A  R      01
BLK_ST_ERR                         000002E9  R      01
BREAK_VC                           0000052F  R      01
BUG$_CIPORT                        ********    X    01
CALL_ERRADDR                       00000698  R      01
CDT$B_RSTATION                  =  00000020
CDT$C_ACCP_PEND                 =  00000002
CDT$C_ACCP_SENT                 =  0000000A
CDT$C_CON_ACK                   =  00000008
CDT$C_CON_PEND                  =  00000001
CDT$C_CON_REC                   =  00000009
CDT$C_CON_SENT                  =  00000007
CDT$C_CR_PEND                   =  00000005
CDT$C_DCR_PEND                  =  00000006
CDT$C_DISC_ACK                  =  00000003
CDT$C_DISC_MTCH                 =  00000006
CDT$C_DISC_PEND                 =  00000004
CDT$C_DISC_REC                  =  00000004
CDT$C_DISC_SENT                 =  00000005
CDT$C_LISTEN                    =  00000001
CDT$C_OPEN                      =  00000002
CDT$C_REJ_PEND                  =  00000003
CDT$C_REJ_SENT                  =  0000000B
CDT$C_VC_FAIL                   =  0000000C
CDT$L_CDTLST                    =  0000006C
CDT$L_CONDAT                    =  00000058
CDT$L_CRWAITQFL                 =  00000038
CDT$L_ERRADDR                   =  0000000C
CDT$L_FPC                       =  00000064
CDT$L_FR5                       =  00000068
CDT$L_LCONID                    =  00000018
CDT$L_LPROCNAM                  =  00000054
CDT$L_MSGINPUT                  =  00000000
CDT$L_PB                        =  0000001C
CDT$L_PDT                       =  00000010
CDT$L_RCONID                    =  00000014
CDT$L_RPROCNAM                  =  00000050
CDT$L_SCSMSG                    =  0000002C
CDT$L_WAITQFL                   =  00000030
CDT$W_BLKSTATE                  =  0000002A
CDT$W_DGREC                     =  0000004C
CDT$W_INITLREC                  =  00000048
CDT$W_MINREC                    =  00000044
CDT$W_MINSEND                   =  0000004A
CDT$W_PENDREC                   =  00000046
CDT$W_REASON                    =  00000026
CDT$W_REC                       =  00000042
CDT$W_SEND                      =  00000040
CDT$W_STATE                     =  00000028
CDT_CLOSED                         000001C5  R      01
CHK_NO_CDTS                        000006D9  R      01
CNF$LKP_PB_MSG2                    ********    X    01
CNF$REMOVE_PB                      ********    X    01
CNF$SCSMSG_REC                     ********    X    01
COMMON_CONNECT                     00000327  R      01
COMMON_RSP1                        000003EB  R      01
COMMON_RSP2                        00000400  R      01
CONFIG_ERR                         0000062A  R      01
CONNECT_ABO                        000006A4  R      01
CONNECT_FAIL                       00000172  R      01
CON_LISTEN                         000000E7  R      01
DCONNECT_OK                        000006BF  R      01
ELOG$PACKET                        ********    X    01
ERR$BUGCHECK                       ********    X    01
ERR$BUGCHECKNF                     ********    X    01
ERR$CLEANUP_PKT                    ********    X    01
ERR$CRASHVC                        ********    X    01
ERR$CRASH_PORT                     ********    X    01
ERR$DEBUGCHECK                     ********    X    01
ERR$INIPORT                        ********    X    01
ERR$V_DEB_CNFER                    ********    X    01
ERR$V_DEB_NOPB                     ********    X    01
FATAL_CR_ERR                       000000AC  R      01
FIND_PRV_CDT                       00000684  R      01
FMT_SCS                            00000313  R      01
FMT_SCS_CREDIT                     00000306  R      01
FPC$CHK_DCONID                     ********    X    01
FPC$INITIAL                        ********    X    01
GET_NEXT_CDT                       000006D0  R      01
IGNORE_CDT                         000006D0  R      01
ILLMSGTYP                          000003CC  R      01
INT$ALLOC_DG                       ********    X    01
INT$ALLOC_MSG                      ********    X    01
INT$CLRCACHE                       ********    X    01
INT$DEAL_DG                        ********    X    01
INT$DEAL_MSG                       ********    X    01
INT$DFQ2POOL                       ********    X    01
INT$INS_DFREEQ                     ********    X    01
INT$INS_MFREEQ                     ********    X    01
INT$MFQ2POOL                       ********    X    01
INT$SNDMSG                         ********    X    01
INT$SNDMSGL                        ********    X    01
INT$TRNMSG                         ********    X    01
MOV_PROCDATA                       00000352  R      01
NEXT_CDT                           0000065B  R      01
NO_MATCH                           0000012F  R      01
NO_MSG_MEM                         000004A0  R      01
NO_RESOURCE                        00000134  R      01
PAER$K_ES_CNPB                  =  00000004
PAER$K_ES_LST0                  =  00000003
PAER$K_ES_LST1                  =  00000009
PAER$K_ES_LST2                  =  00000007
PAER$K_ES_LST3                  =  00000009
PAER$K_ES_LST4                  =  0000000C
PAER$K_ES_NOPB                  =  00000006
PAER$K_ES_SCA                   =  00000005
PAER$K_ET_DALT                  =  00000003
PAER$K_ET_LMLT                  =  00000042
PB$B_RSTATION                   =  0000000C
PB$C_VC_FAIL                    =  00008000
PB$L_CDTLST                     =  00000034
```

```
PB$L_SCSMSG             =  00000040
PB$L_WAITQBL            =  0000003C
PB$L_WAITQFL            =  00000038
PB$W_STATE              =  00000012
PB$W_VCFAIL_RSN         =  00000046
PDT$B_DQIMAP               00000154
PDT$B_HSHUT_DG             00000180
PDT$B_MAX_PORT             0000017C
PDT$B_NXT_PORT             0000017E
PDT$B_PO_LBSTS             00000180
PDT$B_P1_LBSTS             00000181
PDT$B_PLOGMAP              00000134
PDT$B_PORTMAP              00000114
PDT$B_PORT_NUM             0000017D
PDT$B_REQIDPS              0000017F
PDT$C_LENGTH            =  000000E4
PDT$C_PAREGBASE            000000E4
PDT$C_PAREGEND             00000110
PDT$C_PQB               =  000001E0
PDT$L_CNF                  000000E4
PDT$L_CQ0                  000000F0
PDT$L_CQ1                  000000F4
PDT$L_DFQ                  000000FC
PDT$L_DFQHDR               00000208
PDT$L_DGHDRSZ              00000190
PDT$L_DGNETHD              00000194
PDT$L_DQELOGOUT            000002E0
PDT$L_GPTBASE              0000022C
PDT$L_GPTLEN               00000230
PDT$L_LBDG                 00000184
PDT$L_MFQ                  00000100
PDT$L_MFQHDR               0000020C
PDT$L_MQELOGOUT            00000320
PDT$L_MTC                  00000104
PDT$L_PFAR                 00000108
PDT$L_PMC                  000000E8
PDT$L_POLLERDUE            0000018C
PDT$L_POOLDUE              00000188
PDT$L_PPR                  0000010C
PDT$L_PS                   000000EC
PDT$L_PSR                  000000F8
PDT$L_SPTBASE              00000224
PDT$L_SPTLEN               00000228
PDT$L_VBDT                 0000021C
PDT$L_VPQB                 00000218
PDT$Q_COMQ2                000001F0
PDT$Q_COMQ3                000001F8
PDT$Q_COMQBASE             000001E0
PDT$Q_COMQH                000001E8
PDT$Q_COMQL                000001E0
PDT$Q_DFREEQ               000001D0
PDT$Q_FORMPB               00000174
PDT$Q_MFREEQ               000001D8
PDT$Q_RSPQ                 00000200
PDT$Q_TEMP_RSPQ            0000019C
PDT$V_PWF_CLNUP         =  00000000
PDT$W_BDTCEN               00000220
PDT$W_DQELEN               00000210
PDT$W_LPORT_STS            00000110
PDT$W_MQELEN               00000214
PDT$W_PBCOUNT              00000112
PDT$W_STDGDYN              00000198
PDT$W_STDGUSED             0000019A
REC_ACCP_REQ               00000198  R      01
REC_ACCP_RSP               000001CB  R      01
REC_CON_REQ                000000AF  R      01
REC_CON_RSP                0000015C  R      01
REC_CR_REQ                 00000051  R      01
REC_CR_RSP                 0000009C  R      01
REC_DISC_ACK               00000286  R      01
REC_DISC_CLOSED            0000028E  R      01
REC_DISC_OPEN              00000269  R      01
REC_DISC_REQ               00000253  R      01
REC_DISC_RSP               0000029F  R      01
REC_DISC_SENT              00000294  R      01
REC_REJ_REQ                000001FE  R      01
REC_REJ_RSP                0000023B  R      01
REJECT_OK                  000006A9  R      01
RESUME_CONCALL             00000568  R      01
RSP_DISC_MTCH              000002B2  R      01
RSP_DISC_SENT              000002AB  R      01
SCS$ALL_ALLBUF             00000429  RG     01
SCS$ALL_ALLBUF2            0000042E  RG     01
SCS$ALL_DGREC              000004B0  R      01
SCS$ALL_FRDGS              000004AC  RG     01
SCS$ALL_FRMSGS             0000045A  RG     01
SCS$ALL_MSGREC             0000045E  R      01
SCS$ALL_SCSREC             0000044F  R      01
SCS$B_CON_DAT           =  00000024
SCS$CACHECLR               00000712  RG     01
SCS$CHK_SEQNUM             00000575  R      01
SCS$CLOSE_CDT              00000581  RG     01
SCS$COPY_ACCP              00000409  RG     01
SCS$C_ACCP_REQ          =  00000002
SCS$C_ACCP_REQL         =  00000042
SCS$C_ACCP_RSP          =  00000003
SCS$C_ACCP_RSPL         =  00000012
SCS$C_CON_REQ           =  00000000
SCS$C_CON_REQL          =  00000042
SCS$C_CON_RSP           =  00000001
SCS$C_CON_RSPL          =  00000012
SCS$C_CR_REQ            =  00000008
SCS$C_CR_REQL           =  00000012
SCS$C_CR_RSP            =  00000009
SCS$C_CR_RSPL           =  0000000E
SCS$C_DISC_REQ          =  00000006
SCS$C_DISC_REQL         =  00000012
SCS$C_DISC_RSP          =  00000007
SCS$C_DISC_RSPL         =  0000000E
SCS$C_OVHD              =  0000000E
SCS$C_REJ_REQ           =  00000004
SCS$C_REJ_REQL          =  00000012
SCS$C_REJ_RSP           =  00000005
SCS$C_REJ_RSPL          =  0000000E
```

```
SCS$C_STDISC                    =  00000019
SCS$C_STINSFCR                  =  00000021
SCS$C_STNOMAT                   =  0000000A
SCS$C_STNORMAL                  =  00000001
SCS$C_STNORS                    =  00000012
SCS$DEALL_CDT                      ********    X   01
SCS$DEAL_ALLBUF                    000004F2 RG     01
SCS$DEAL_DGREC                     0000050C R      01
SCS$DEAL_MSGREC                    000004F6 R      01
SCS$DEAL_SCSREC                    0000051D RG     01
SCS$DISC_VCFAIL                    000005D1 RG     01
SCS$FREE_LISTEN                    0000053F RG     01
SCS$GL_SCSSIZE                     00000022 RG     01
SCS$INITIAL                        00000026 RG     01
SCS$LOCLOOKUP                      ********    X   01
SCS$L_DST_CONID                 =  FFFFFFF8
SCS$L_SRC_CONID                 =  FFFFFFFC
SCS$MAP_SCSSTS                     000005A0 R      01
SCS$MAP_VMSSTS                     000005BA RG     01
SCS$NOTIFY_SYSAP                   00000648 RG     01
SCS$REC_SCSMSG                     00000029 RG     01
SCS$REQ_SCSSEND                    000002BB RG     01
SCS$RESUMEWAITR                    ********    X   01
SCS$RESUM_SEND                     00000396 R      01
SCS$SAVE_REQ                       00000418 R      01
SCS$SEND_RSP                       000003B9 R      01
SCS$SETCRT_CLSD                    00000603 RG     01
SCS$T_DST_PROC                  =  00000004
SCS$T_SRC_PROC                  =  00000014
SCS$VCCLOSED                       000005F7 RG     01
SCS$W_CREDIT                    =  FFFFFFF6
SCS$W_LENGTH                    =  FFFFFFF0
SCS$W_MIN_CR                    =  00000000
SCS$W_MTYPE                     =  FFFFFFF4
SCS$W_STATUS                    =  00000002
SCS_STATUS_LEN                  =  00000012
SCS_STATUS_TAB                     00000000 R      01
SEND_ACCEPT                        0000035C R      01
SEND_ACCP_RSP                      000003D3 R      01
SEND_CONNECT                       0000031A R      01
SEND_CON_RSP                       000003DF R      01
SEND_CREDIT                        000002F0 R      01
SEND_CR_RSP                        000003E7 R      01
SEND_DISCONNECT                    0000037D R      01
SEND_DISC_RSP                      000003D9 R      01
SEND_NEXT_SEND                     000002D4 R      01
SEND_REJECT                        0000036B R      01
SEND_REJ_RSP                       000003D9 R      01
SIZ...                          =  00000001
SNDMSG                             00000349 R      01
SS$_DISCONNECT                  =  0000204C
SS$_INSFMEM                     =  00000124
SS$_NOLISTENER                  =  0000215C
SS$_NORMAL                      =  00000001
SS$_REJECT                      =  00000294
SS$_REMRSRC                     =  0000206C
SS$_VCBROKEN                    =  0000219C
UNLNK_CDT                          00000690 R      01
UPDATE_SEND                        0000007A R      01
VC_OPEN                            000000BC R      01
VMS_STATUS_TAB                     00000012 R      01
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
$$$115_DRIVER	00000738 (1848.)	01 (1.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
ABS	00000360 (864.)	02 (2.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	30	00:00:00.04	00:00:00.89
Command processing	110	00:00:00.51	00:00:04.35
Pass 1	414	00:00:11.76	00:00:42.60
Symbol table sort	0	00:00:01.15	00:00:05.66
Pass 2	397	00:00:04.04	00:00:12.95
Symbol table output	1	00:00:00.16	00:00:00.27
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	954	00:00:17.68	00:01:06.88

The working set limit was 2100 pages.
96016 bytes (188 pages) of virtual memory were used to buffer the intermediate code.
There were 60 pages of symbol table space allocated to hold 1042 non-local and 92 local symbols.
2657 source lines were read in Pass 1, producing 22 object records in Pass 2.
33 pages of virtual memory were used to define 31 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[DRIVER.OBJ]PALIB.MLB;1	7
_255DUA28:[SYS.OBJ]LIB.MLB;1	8
_255DUA28:[SYSLIB]STARLET.MLB;2	7
TOTALS (all libraries)	22

1238 GETS were required to define 22 macros.

There were no errors, warnings or information messages.

```
MACRO/LIS=LIS$:PASCSCTL/OBJ=OBJ$:PASCSCTL MSRC$:PASCSCTL/UPDATE=(ENH$:PASCSCTL)+EXECML$/LIB+LIB$:PALIB.MLB/LIB
```

PUDRIVER
LIS

RTTDRIVER
LIS

PATABLES
LIS

PASCSCTL
LIS