PPPPPPPPPPPP RRRRRRRRRRR TTTTTTTTTTTTTTT SSSSSSSSSSS MMM MMM BBBBBBBBBBB
PPPPPPPPPPPP RRRRRRRRRRR TTTTTTTTTTTTTTT SSSSSSSSSSS MMM MMM BBBBBBBBBBB
PPPPPPPPPPPP RRRRRRRRRRR TTTTTTTTTTTTTTT SSSSSSSSSSS MMM MMM BBBBBBBBBBB
PPP PPP RRR RRR TTT SSS MMMMMM MMMMMM BBB BBB
PPP PPP RRR RRR TTT SSS MMMMMM MMMMMM BBB BBB
PPP PPP RRR RRR TTT SSS MMMMMM MMMMMM BBB BBB
PPP PPP RRR RRR TTT SSS MMM MMM MMM BBB BBB
PPP PPP RRR RRR TTT SSS MMM MMM MMM BBB BBB
PPP PPP RRR RRR TTT SSS MMM MMM MMM BBB BBB
PPPPPPPPPPPP RRRRRRRRRRR TTT SSSSSSSSS MMM MMM BBBBBBBBBBB
PPPPPPPPPPPP RRRRRRRRRRR TTT SSSSSSSSS MMM MMM BBBBBBBBBBB
PPPPPPPPPPPP RRRRRRRRRRR TTT SSSSSSSSS MMM MMM BBBBBBBBBBB
PPP RRR RRR TTT SSS MMM MMM BBB BBB
PPP RRR RRR TTT SSS MMM MMM BBB BBB
PPP RRR RRR TTT SSS MMM MMM BBB BBB
PPP RRR RRR TTT SSS MMM MMM BBB BBB
PPP RRR RRR TTT SSS MMM MMM BBB BBB
PPP RRR RRR TTT SSSSSSSSSSS MMM MMM BBBBBBBBBBB
PPP RRR RRR TTT SSSSSSSSSSS MMM MMM BBBBBBBBBBB
PPP RRR RRR TTT SSSSSSSSSSS MMM MMM BBBBBBBBBBB
PPP RRR RRR TTT SSSSSSSSSSS MMM MMM BBBBBBBBBBB

```
  SSSSSSSS  EEEEEEEEEE  RRRRRRRR    VV      VV    IIIIII      CCCCCCCC  EEEEEEEEEE    SSSSSSSS
  SSSSSSSS  EEEEEEEEEE  RRRRRRRR    VV      VV    IIIIII      CCCCCCCC  EEEEEEEEEE    SSSSSSSS
SS          EE          RR      RR  VV      VV      II      CC          EE          SS
SS          EE          RR      RR  VV      VV      II      CC          EE          SS
SS          EE          RR      RR  VV      VV      II      CC          EE          SS
SS          EE          RR      RR  VV      VV      II      CC          EE          SS
  SSSSSS    EEEEEEEE    RRRRRRRR    VV      VV      II      CC          EEEEEEEE      SSSSSS
  SSSSSS    EEEEEEEE    RRRRRRRR    VV      VV      II      CC          EEEEEEEE      SSSSSS
        SS  EE          RR  RR      VV      VV      II      CC          EE                  SS
        SS  EE          RR  RR      VV      VV      II      CC          EE                  SS
        SS  EE          RR    RR      VV  VV        II      CC          EE                  SS      ....
        SS  EE          RR    RR      VV  VV        II      CC          EE                  SS      ....
SSSSSSSS    EEEEEEEEEE  RR      RR      VV        IIIIII      CCCCCCCC  EEEEEEEEEE  SSSSSSSS        ....
SSSSSSSS    EEEEEEEEEE  RR      RR      VV        IIIIII      CCCCCCCC  EEEEEEEEEE  SSSSSSSS        ....

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
```

```
;     1          0001  0 MODULE SERVICES (%TITLE 'Symbiont Services -- Shareable Routines'
;     2          0002  0                   IDENT = 'V04-000',
;     3          0003  0                   ADDRESSING_MODE (EXTERNAL = GENERAL)
;     4          0004  0                   ) =
;     5          0005  1 BEGIN
;     6          0006  1
;     7          0007  1 !
;     8          0008  1 !*****************************************************************************
;     9          0009  1 !*                                                                           *
;    10          0010  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;    11          0011  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    12          0012  1 !*  ALL RIGHTS RESERVED.                                                      *
;    13          0013  1 !*                                                                           *
;    14          0014  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
;    15          0015  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
;    16          0016  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
;    17          0017  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
;    18          0018  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
;    19          0019  1 !*  TRANSFERRED.                                                              *
;    20          0020  1 !*                                                                           *
;    21          0021  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
;    22          0022  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT     *
;    23          0023  1 !*  CORPORATION.                                                              *
;    24          0024  1 !*                                                                           *
;    25          0025  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
;    26          0026  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;    27          0027  1 !*                                                                           *
;    28          0028  1 !*                                                                           *
;    29          0029  1 !*****************************************************************************
;    30          0030  1 !
;    31          0031  1
;    32          0032  1 !++
;    33          0033  1 ! FACILITY:
;    34          0034  1 !       Symbiont Services
;    35          0035  1 !
;    36          0036  1 ! ABSTRACT:
;    37          0037  1 !       This module provides a set of shareable routines used by symbionts.
;    38          0038  1 !
;    39          0039  1 ! ENVIRONMENT:
;    40          0040  1 !       VAX/VMS user mode.
;    41          0041  1 !--
;    42          0042  1 !
;    43          0043  1 ! AUTHOR: Greg Robert, CREATION DATE: 26-Apr-1983
;    44          0044  1 !
;    45          0045  1 ! MODIFIED BY:
;    46          0046  1 !
;    47          0047  1 !       V3-008  RRB0014         Rowland R. Bradley      09-Aug-1984
;    48          0048  1 !               Change the allocation of the IOB from 512 to 1024.
;    49          0049  1 !               Previous to this change long filenames would cause
;    50          0050  1 !               the username and UIC to disappear.  The long filename
;    51          0051  1 !               was passed by the job controller in a message of 1024.
;    52          0052  1 !               The print symbionts largest message buffer (IOB) was 512.
;    53          0053  1 !               This change increases the IOB to 1024.
;    54          0054  1 !
;    55          0055  1 !       V3-007  RRB0013         Rowland R. Bradley      20-Jul-1984
;    56          0056  1 !               Change messages PSM$_NOMOREITEMS to SMB$_NOMOREITEMS,
;    57          0057  1 !               PSM$_INVSTRLEV to SMB$_INVSTRLEV, PSM$_INVSTMNBR to
```

```
;    58          0058  1 !                SMB$_INVSTMSBR.
;    59          0059  1 !
;    60          0060  1 !       V3-006  GRR0012         Gregory R. Robert       21-Oct-1983
;    61          0061  1 !               Disable user-allowed check in provide_service
;    62          0062  1 !
;    63          0063  1 !       V3-005  GRR0011         Gregory R. Robert       21-OCT-1983
;    64          0064  1 !               Remove user argument optional parameter from provide_service.
;    65          0065  1 !
;    66          0066  1 !       V3-004  GRR0010         Gregory R. Robert       18-Oct-1983
;    67          0067  1 !               Fix improper bind statement in provide_service
;    68          0068  1 !
;    69          0069  1 !       3B-003  GRR3003         Gregory R. Robert       23-Aug-1983
;    70          0070  1 !               Bugfixes, page_setup_modules, form_setup_modules,
;    71          0071  1 !               sheet_feed, symbiont initiated pause_task and stop_stream,
;    72          0072  1 !               hangup code, read and write item services
;    73          0073  1 !
;    74          0074  1 !       3B-002  GRR3002         Gregory R. Robert       03-Aug-1983
;    75          0075  1 !               Rewrite for new design.
;    76          0076  1 !
;    77          0077  1 !       3B-001  GRR4003         Gregory R. Robert       29-Jul-1983
;    78          0078  1 !               Created new module.
;    79          0079  1 !
;    80          0080  1 !**
```

```
;    82          0081  1
;    83          0082  1 LIBRARY 'SYS$LIBRARY:LIB';
;    84          0083  1 REQUIRE 'LIB$:SMBDEF';
;    85          0575  1 REQUIRE 'SRC$:SMBREQ';
;    86          1032  1
;    87          1033  1 EXTERNAL ROUTINE
;    88          1034  1         PSM$ALLOCATE_DSB:               NOVALUE,
;    89          1035  1         PSM$ALLOCATE_IOB:               NOVALUE,
;    90          1036  1         PSM$DEALLOCATE_DSB:             NOVALUE
;    91          1037  1         ;
;    92          1038  1
;    93          1039  1 EXTERNAL
;    94          1040  1         PSM$SRV : BLOCKVECTOR[,SRV_S_SRV, BYTE]
;    95          1041  1         ;
;    96          1042  1
;    97          1043  1
;    98          1044  1 FORWARD ROUTINE
;    99          1045  1         SMB$CHECK_FOR_MESSAGE    ,
;   100          1046  1         SMB$READ_MESSAGE_ITEM    ,
;   101          1047  1         SMB$INITIALIZE           ,
;   102          1048  1         SMB$READ_MESSAGE         ,
;   103          1049  1         SMB$SEND_TO_JOBCTL       ,
;   104          1050  1
;   105          1051  1         PSM$REPLACE              ,
;   106          1052  1         PSM$SCHEDULE_NON_AST     ,
;   107          1053  1         PSM$WAIT_FOR_NON_AST     ,
;   108          1054  1
;   109          1055  1         READ_MAILBOX             : NOVALUE,
;   110          1056  1         READ_MAILBOX_AST         : NOVALUE,
;   111          1057  1         WRITE_MAILBOX            : NOVALUE
;   112          1058  1         ;
;   113          1059  1
;   114          1060  1 OWN
;   115          1061  1         NON_AST_QUEUE:  VECTOR [2],          ! Non-ast queue header
;   116          1062  1         MESSAGE_QUEUE:  VECTOR [2],          ! Message queue header
;   117          1063  1         MESSAGE_AST_ROUTINE,                 ! User AST routine
;   118          1064  1         MAXIMUM_STREAMS,                     ! User specified stream limit
;   119          1065  1         IMBX_CHAN:      WORD,                ! Input mailbox
;   120          1066  1         OMBX_CHAN:      WORD,                ! Output mailbox channel
;   121          1067  1         OMBX_IOSB:      VECTOR [4, WORD]     ! Output mailbox iosb
;   122          1068  1         ;
```

```
;  124          1069  1 GLOBAL ROUTINE SMB$CHECK_FOR_MESSAGE %SBTTL 'CHECK_FOR_MESSAGE'
;  125          1070  1         =
;  126          1071  2 BEGIN
;  127          1072  2 !++
;  128          1073  2 !
;  129          1074  2 ! FUNCTIONAL DESCRIPTION:
;  130          1075  2 !       Checks if the message queue is empty.
;  131          1076  2 !
;  132          1077  2 ! INPUT PARAMETERS:
;  133          1078  2 !       NONE
;  134          1079  2 !
;  135          1080  2 ! IMPLICIT INPUTS:
;  136          1081  2 !       <tbs>           - Message queue header.
;  137          1082  2 !
;  138          1083  2 ! OUTPUT PARAMETERS:
;  139          1084  2 !       NONE
;  140          1085  2 !
;  141          1086  2 ! IMPLICIT OUTPUTS:
;  142          1087  2 !       NONE
;  143          1088  2 !                                                                                ;
;  144          1089  2 ! ROUTINE VALUE:
;  145          1090  2 !       SS$_NORMAL      - A message is waiting.
;  146          1091  2 !       <tbs>           - The message queue is empty.
;  147          1092  2 !
;  148          1093  2 ! SIDE EFFECTS:
;  149          1094  2 !       NONE
;  150          1095  2 !
;  151          1096  2 !--
;  152          1097  2
;  153          1098  2 IF .MESSAGE_QUEUE [0] EQL MESSAGE_QUEUE [0]
;  154          1099  2 THEN
;  155          1100  2     RETURN 0
;  156          1101  2 ELSE
;  157          1102  2     RETURN SS$_NORMAL;
;  158          1103  2
;  159          1104  1 END;


                                        .TITLE  SERVICES Symbiont Services -- Shareable Routine
;                                       .IDENT  \V04-000\                                        s

                                        .PSECT  DATA,NOEXE,2

                            00000 NON_AST_QUEUE:
                                        .BLKB   8
                            00008 MESSAGE_QUEUE:
                                        .BLKB   8
                            00010 MESSAGE_AST_ROUTINE:
                                        .BLKB   4
                            00014 MAXIMUM_STREAMS:
                                        .BLKB   4
                            00018 IMBX_CHAN:
                                        .BLKB   2
                            0001A OMBX_CHAN:
                                        .BLKB   2
                            0001C OMBX_IOSB:
```

```
                                                .BLKB   8

                                                .EXTRN  BAS$EDIT, LBR$CLOSE
                                                .EXTRN  LBR$GET_RECORD, LBR$INI_CONTROL
                                                .EXTRN  LBR$LOOKUP_KEY, LBR$OPEN
                                                .EXTRN  LBR$RET_RMSSTV, LBR$SET_LOCATE
                                                .EXTRN  LIB$TRIM_FILESPEC
                                                .EXTRN  LIB$GET_VM, LIB$FREE_VM
                                                .EXTRN  STR$ANALYZE_SDESC
                                                .EXTRN  STR$ANALYZE_SDESC_R1
                                                .EXTRN  STR$APPEND, STR$CONCAT
                                                .EXTRN  STR$COPY_DX, STR$COPY_R
                                                .EXTRN  STR$FREE1_DX, STR$FREE1_DX_R4
                                                .EXTRN  STR$GET1_DX, STR$LEFT
                                                .EXTRN  STR$PREFIX, STR$RIGHT
                                                .EXTRN  PSM$S_HANGUP_DISPATCH_ENTRY
                                                .EXTRN  PSM$_BUFFEROVF, PSM$_EOF
                                                .EXTRN  PSM$_ESCAPE, PSM$_FLUSH
                                                .EXTRN  PSM$_FUNNOTSUP, PSM$_INVITMCOD
                                                .EXTRN  PSM$_INVVMSOSC, PSM$_MODNOTFND
                                                .EXTRN  PSM$_NEWPAGE, PSM$_NOFILEID
                                                .EXTRN  PSM$_OSCTOOLON, PSM$_PENDING
                                                .EXTRN  PSM$_SUSPEND, PSM$_TOOMANYLEV
                                                .EXTRN  SMB$_INVSTMNBR, SMB$_INVSTRLEV
                                                .EXTRN  SMB$_NOMOREITEMS
                                                .EXTRN  PSM$ALLOCATE_DSB
                                                .EXTRN  PSM$ALLOCATE_IOB
                                                .EXTRN  PSM$DEALLOCATE_DSB
                                                .EXTRN  PSM$SRV

                                                .PSECT  CODE,NOWRT,2

                                  0000 00000    .ENTRY  SMB$CHECK_FOR_MESSAGE, Save nothing     ; 1069
50      0000'   CF   9E 00002                  MOVAB   MESSAGE_QUEUE, R0                       ; 1098
50      0000'   CF   D1 00007                  CMPL    MESSAGE_QUEUE, R0                       ;
                03   12 0000C                  BNEQ    1$                                      ;
                50   D4 0000E                  CLRL    R0                                      ; 1102
                     04 00010                  RET                                             ;
50              01   D0 00011 1$:              MOVL    #1, R0                                  ;
                     04 00014                  RET                                             ; 1104

; Routine Size:  21 bytes,    Routine Base:  CODE + 0000
```

```
;  161          1105  1 GLOBAL ROUTINE SMB$READ_MESSAGE_ITEM ( %SBTTL 'GET_NEXT_ITEM'
;  162          1106  1          MESSAGE ,
;  163          1107  1          CONTEXT ,
;  164          1108  1          ITEM    : REF $LONGWORD,
;  165          1109  1          BUFFER  ,
;  166          1110  1          SIZE    : REF $WORD
;  167          1111  1          ) =
;  168          1112  2 BEGIN
;  169          1113  2          PARAMETER_INDEX_ (MESSAGE, CONTEXT, ITEM, BUFFER, SIZE);
;  170          1114  2
;  171          1115  2 !++
;  172          1116  2 !
;  173          1117  2 ! FUNCTIONAL DESCRIPTION:
;  174          1118  2 !          Parses and returns the next item in the message.
;  175          1119  2 !
;  176          1120  2 ! INPUT PARAMETERS:
;  177          1121  2 !          MESSAGE          - Descriptor of message buffer.
;  178          1122  2 !          CONTEXT          - Longword context, next item pointer.
;  179          1123  2 !          ITEM             - Longword to receive item code.
;  180          1124  2 !          BUFFER           - Descriptor of buffer to receive item.
;  181          1125  2 !          SIZE             - Word to receive item size.
;  182          1126  2 !
;  183          1127  2 ! IMPLICIT INPUTS:
;  184          1128  2 !          NONE
;  185          1129  2 !
;  186          1130  2 ! OUTPUT PARAMETERS:
;  187          1131  2 !          NONE
;  188          1132  2 !
;  189          1133  2 ! IMPLICIT OUTPUTS:
;  190          1134  2 !          NONE
;  191          1135  2 !
;  192          1136  2 ! ROUTINE VALUE:
;  193          1137  2 !          SS$_NORMAL       - Normal successful completion.
;  194          1138  2 !          LIB$_INVARG      - Invalid argument.
;  195          1139  2 !          SMB$_NOMOREITEMS- Item list exhausted.
;  196          1140  2 !          ...              - Any return from LIB$SCOPY_xxxx routines.
;  197          1141  2 !
;  198          1142  2 ! SIDE EFFECTS:
;  199          1143  2 !          Context field updated to point at next item or cleared to
;  200          1144  2 !          zero when last item +1 is read.
;  201          1145  2 !
;  202          1146  2 !--
;  203          1147  2
;  204          1148  2 LOCAL
;  205          1149  2          MSG_ADDRESS,
;  206          1150  2          MSG_SIZE:        INITIAL (0)
;  207          1151  2          ;
;  208          1152  2
;  209          1153  2 BIND
;  210          1154  2          I_PTR = .CONTEXT : REF $BBLOCK
;  211          1155  2          ;
;  212          1156  2
;  213          1157  2
;  214          1158  2 ! Get the size and address of the message.
;  215          1159  2 !
;  216          1160  2 STR$ANALYZE_SDESC (.MESSAGE, MSG_SIZE, MSG_ADDRESS);
;  217          1161  2
```

```
;  218          1162  2
;  219          1163  2 ! Advance to next item, or first item if starting.
;  220          1164  2 !
;  221          1165  2 IF .I_PTR EQL 0
;  222          1166  2 THEN
;  223          1167  2     I_PTR = .MSG_ADDRESS + SMBMSG$S_REQUEST_HEADER
;  224          1168  2 ELSE
;  225          1169  2     I_PTR = .I_PTR +  .I_PTR[SMBMSG$W_ITEM_SIZE] + SMBMSG$S_ITEM_HEADER;
;  226          1170  2
;  227          1171  2
;  228          1172  2 ! If end of message (item code eql 0) or past end of message buffer
;  229          1173  2 ! then reset context and return no more items.
;  230          1174  2 !
;  231          1175  3 IF .I_PTR[SMBMSG$W_ITEM_CODE] EQL 0 OR .I_PTR GTRA (.MSG_ADDRESS + .MSG_SIZE)
;  232          1176  2 THEN
;  233          1177  3     BEGIN
;  234          1178  3     I_PTR = 0;
;  235          1179  3     RETURN SMB$_NOMOREITEMS;
;  236          1180  2     END;
;  237          1181  2
;  238          1182  2
;  239          1183  2 ! Return item code and item size if requested.
;  240          1184  2 !
;  241          1185  2 ITEM[] = .I_PTR[SMBMSG$W_ITEM_CODE];
;  242          1186  2 IF PARAMETER_PRESENT_ (SIZE)
;  243          1187  2 THEN
;  244          1188  2     SIZE[] = .I_PTR[SMBMSG$W_ITEM_SIZE];
;  245          1189  2
;  246          1190  2
;  247          1191  2 ! Copy the item to the user buffer
;  248          1192  2 !
;  249          1193  2 COPY_R_DX_ (I_PTR[SMBMSG$W_ITEM_SIZE], .I_PTR + SMBMSG$S_ITEM_HEADER, .BUFFER);
;  250          1194  2
;  251          1195  2 SS$_NORMAL
;  252          1196  2
;  253          1197  1 END;


                                  0004 00000          .ENTRY  SMB$READ_MESSAGE_ITEM, Save R2                  ; 1105
                          5E        08  C2 00002          SUBL2   #8, SP                                          ;
                                04  AE  D4 00005          CLRL    MSG_SIZE                                        ; 1112
                          52    08  AC  D0 00008          MOVL    CONTEXT, R2                                     ; 1154
                                5E  DD 0000C              PUSHL   SP                                              ; 1160
                                08  AE  9F 0000E          PUSHAB  MSG_SIZE                                        ;
                                04  AC  DD 00011          PUSHL   MESSAGE                                         ;
                   00000000G    00       03  FB 00014    CALLS   #3, STR$ANALYZE_SDESC                           ;
                                         62  D5 0001B          TSTL    (R2)                                        ; 1165
                                         06  12 0001D          BNEQ    1$                                          ;
                          62            6E       04  C1 0001F    ADDL3   #4, MSG_ADDRESS, (R2)                   ; 1167
                                         0B  11 00023          BRB     2$                                          ;
                          50    00  B2  3C 00025 1$:      MOVZWL  @0(R2), R0                                      ; 1169
                          50        62  C0 00029          ADDL2   (R2), R0                                        ;
                          62    04  A0  9E 0002C          MOVAB   4(R0), (R2)                                     ;
                          50        62  D0 00030 2$:      MOVL    (R2), R0                                        ; 1175
```

```
                                    02    A0  B5 00033          TSTW    2(R0)                                   ;
                                    0A    13  00036          BEQL    3$                                      ;
                 51          6E     04    AE  C1 00038          ADDL3   MSG_SIZE, MSG_ADDRESS, R1            ;
                             51           50  D1 0003D          CMPL    R0, R1                                  ;
                                          0A  1B 00040          BLEQU   4$                                      ;
                                          62  D4 00042 3$:      CLRL    (R2)                                    ; 1178
                             50 00000000G 8F  D0 00044          MOVL    #SMB$_NOMOREITEMS, R0                   ; 1179
                                          04  00048          RET                                             ;
                             50           62  D0 0004C 4$:      MOVL    (R2), R0                                ; 1185
                      0C     BC     02    A0  3C 0004F          MOVZWL  2(R0), @ITEM                            ;
                             05           6C  91 00054          CMPB    (AP), #5                                ; 1186
                                          09  1F 00057          BLSSU   5$                                      ;
                                    14    AC  D5 00059          TSTL    20(AP)                                  ;
                                          04  13 0005C          BEQL    5$                                      ;
                      14     BC           60  B0 0005E          MOVW    (R0), @SIZE                             ; 1188
                                    04    A0  9F 00062 5$:      PUSHAB  4(R0)                                   ; 1193
                                          50  DD 00065          PUSHL   R0                                      ;
                                    10    AC  DD 00067          PUSHL   BUFFER                                  ;
                 00000000G           00     03  FB 0006A          CALLS   #3, STR$COPY_R                          ;
                             52           50  D0 00071          MOVL    R0, STATUS                              ;
                             09           52  E8 00074          BLBS    STATUS, 6$                              ;
                                          52  DD 00077          PUSHL   STATUS                                  ;
                 00000000G           00     01  FB 00079          CALLS   #1, LIB$SIGNAL                          ;
                             50           01  D0 00080 6$:      MOVL    #1, R0                                  ; 1197
                                          04  00083          RET                                             ;

; Routine Size:  132 bytes,    Routine Base:  CODE + 0015
```

```
;  255          1198  1 ! Functional Description:
;  256          1199  1 !                 ?desc
;  257          1200  1 !
;  258          1201  1 ! Formal Parameters:
;  259          1202  1 !                 ?desc
;  260          1203  1 ! Implicit Inputs:
;  261          1204  1 !                 none
;  262          1205  1 !
;  263          1206  1 ! Implicit Outputs:
;  264          1207  1 !                 none
;  265          1208  1 !
;  266          1209  1 ! Returned Value:
;  267          1210  1 !                 none
;  268          1211  1 !
;  269          1212  1 ! Side Effects:
;  270          1213  1 !                 none
;  271          1214  1 !--
;  272          1215  1 GLOBAL ROUTINE PSM$REPLACE ( %SBTTL 'PROVIDE_SERVICE'
;  273          1216  1         SERVICE_CODE    : REF $LONGWORD,
;  274          1217  1         RTNADR
;  275          1218  1         ) =
;  276          1219  2 BEGIN
;  277          1220  2 BIND SERVICE = PSM$SRV [.SERVICE_CODE[], 0,0,0,0] : $BBLOCK;
;  278          1221  2
;  279          1222  2 ! Validate the service code
;  280          1223  2 !
;  281          1224  2 IF .SERVICE_CODE[] EQL 0
;  282          1225  2 OR .SERVICE_CODE[] GEQU PSM$K_MAX
;  283          1226  2 THEN
;  284          1227  2     CODEERR_ ;
;  285          1228  2
;  286          1229  2
;  287          1230  2 ! Check that this service may be user supplied
;  288          1231  2 !
;  289          1232  2 !*!Allow any user supplied service for FT1
;  290          1233  2 !*!
;  291          1234  2 !*!IF NOT .SERVICE[SRV_V_USER_ALLOWED]
;  292          1235  2 !*!THEN
;  293          1236  2 !*!     CODEERR_ ;
;  294          1237  2
;  295          1238  2
;  296          1239  2 ! Update the table with the routine address
;  297          1240  2 !
;  298          1241  2 SERVICE[SRV_A_SERVICE] = .RTNADR;
;  299          1242  2
;  300          1243  2
;  301          1244  2 ! Mark the service as user supplied
;  302          1245  2 !
;  303          1246  2 SERVICE[SRV_V_USER_SUPPLIED] = 1;
;  304          1247  2
;  305          1248  2 SS$_NORMAL
;  306          1249  2
;  307          1250  1 END;
```

```
                                                0004 00000          .ENTRY  PSM$REPLACE, Save R2                    ; 1215
                50        04    BC                04   78 00002          ASHL    #4, @SERVICE_CODE, R0           ; 1220
                                52 00000000G0040    9E 00007          MOVAB   PSM$SRV[R0], R2                 ;
                                        04    BC    D5 0000F          TSTL    @SERVICE_CODE                   ; 1224
                                              06    13 00012          BEQL    1$                              ;
                                17      04    BC    D1 00014          CMPL    @SERVICE_CODE, #23              ; 1225
                                              0F    1F 00018          BLSSU   2$                              ;
                                              01    DD 0001A 1$:      PUSHL   #1                              ; 1226
                                   01061154   8F    DD 0001C          PUSHL   #17174868                       ;
                    00000000G   00           02    FB 00022          CALLS   #2, LIB$STOP                    ;
                                62      08    AC    D0 00029 2$:      MOVL    RTNADR, (R2)                    ; 1241
                          08    A2            02    88 0002D          BISB2   #2, 8(R2)                       ; 1246
                                50            01    D0 00031          MOVL    #1, R0                          ; 1250
                                              04    00034             RET                                     ;

; Routine Size:  53 bytes,    Routine Base:  CODE + 0099
```

```
;  309          1251  1 GLOBAL ROUTINE SMB$INITIALIZE ( %SBTTL 'INITIALIZE_SYMBIONT'
;  310          1252  1         STRUCTURE_LEVEL : REF $LONGWORD,
;  311          1253  1         AST_ROUTINE     ,
;  312          1254  1         STREAMS         : REF $LONGWORD
;  313          1255  1         ) =
;  314          1256  2 BEGIN
;  315          1257  2         PARAMETER_INDEX_ (STRUCTURE_LEVEL, AST_ROUTINE, STREAMS);
;  316          1258  2
;  317          1259  2 !++
;  318          1260  2 !
;  319          1261  2 ! FUNCTIONAL DESCRIPTION:
;  320          1262  2 !       Initializes the internal symbiont database and establishes
;  321          1263  2 !       communications with the controlling process.
;  322          1264  2 !
;  323          1265  2 ! INPUT PARAMETERS:
;  324          1266  2 !       STRUCTURE_LEVEL - Address of a longword containing structure level of caller.
;  325          1267  2 !       AST_ROUTINE     - Address of the entry mask of an AST routine
;  326          1268  2 !                         to be called when messages are received.
;  327          1269  2 !       STREAMS         - Longword containing the maximum number of
;  328          1270  2 !                         streams allowed for this symbiont, default 1.
;  329          1271  2 !
;  330          1272  2 ! IMPLICIT INPUTS:
;  331          1273  2 !       <tbs>           - Symbiont database.
;  332          1274  2 !       SYS$INPUT       - Input mailbox from controlling process.
;  333          1275  2 !       SYS$OUTPUT      - Output mailbox to controlling process.
;  334          1276  2 !
;  335          1277  2 ! OUTPUT PARAMETERS:
;  336          1278  2 !       NONE
;  337          1279  2 !
;  338          1280  2 ! IMPLICIT OUTPUTS:
;  339          1281  2 !       NONE
;  340          1282  2 !
;  341          1283  2 ! ROUTINE VALUE:
;  342          1284  2 !       SS$_NORMAL      - Normal, successful completion.
;  343          1285  2 !       LIB$_INVARG     - Invalid arguments.
;  344          1286  2 !       ...             - Any system service failure.
;  345          1287  2 !
;  346          1288  2 ! SIDE EFFECTS:
;  347          1289  2 !       Symbiont database initialized, mailbox channels assigned,
;  348          1290  2 !       AST mailbox read posted.
;  349          1291  2 !
;  350          1292  2 !--
;  351          1293  2
;  352          1294  2 LOCAL
;  353          1295  2         IOB
;  354          1296  2         ;
;  355          1297  2
;  356          1298  2 IF .STRUCTURE_LEVEL[] NEQ SMBMSG$K_STRUCTURE_LEVEL
;  357          1299  2 THEN
;  358          1300  2     RETURN SMB$_INVSTRLEV;
;  359          1301  2
;  360          1302  2 RETURN_IF_ERROR_ ($ASSIGN (DEVNAM=$DESCRIPTOR ('SYS$INPUT'), CHAN=IMBX_CHAN));
;  361          1303  2
;  362          1304  2 RETURN_IF_ERROR_ ($ASSIGN (DEVNAM=$DESCRIPTOR ('SYS$OUTPUT'), CHAN=OMBX_CHAN));
;  363          1305  2
;  364          1306  2 INIT_QUEUE_HEADER_ (NON_AST_QUEUE[0]);
;  365          1307  2 INIT_QUEUE_HEADER_ (MESSAGE_QUEUE[0]);
```

```
;  366      1308  2
;  367      1309  2 IF PARAMETER_PRESENT_ (AST_ROUTINE)
;  368      1310  2 THEN
;  369      1311  2     MESSAGE_AST_ROUTINE = .AST_ROUTINE;
;  370      1312  2
;  371      1313  2 MAXIMUM_STREAMS = 1;
;  372      1314  2 IF PARAMETER_PRESENT_ (STREAMS)
;  373      1315  2 THEN
;  374      1316  2     MAXIMUM_STREAMS = .STREAMS[];
;  375      1317  2
;  376      1318  2 RETURN_IF_ERROR_ ($PURGWS (INADR=UPLIT (0, %X'7FFFFFFF')));
;  377      1319  2
;  378      1320  2 PSM$ALLOCATE_IOB (IOB, UPLIT (1024));
;  379      1321  2 READ_MAILBOX (.IOB);
;  380      1322  2
;  381      1323  2 SS$_NORMAL
;  382      1324  2
;  383      1325  1 END;


         54 55 50 4E 49 24 53 59 53  000CE P.AAB:  .ASCII  \SYS$INPUT\                         ;
                                     000D7         .BLKB   1
                           00000009  000D8 P.AAA:  .LONG   9                                   ;
                           00000000' 000DC         .ADDRESS P.AAB                              ;
      54 55 50 54 55 4F 24 53 59 53  000E0 P.AAD:  .ASCII  \SYS$OUTPUT\                        ;
                                     000EA         .BLKB   2
                           0000000A  000EC P.AAC:  .LONG   10                                  ;
                           00000000' 000F0         .ADDRESS P.AAD                              ;
                  7FFFFFFF 00000000  000F4 P.AAE:  .LONG   0, 2147483647                       ;
                           00000400  000FC P.AAF:  .LONG   1024                                ;

                                                   .EXTRN  SYS$ASSIGN, SYS$PURGWS

                              000C 00000           .ENTRY  SMB$INITIALIZE, Save R2,R3          ; 1251
              53 00000000G  00  9E 00002           MOVAB   SYS$ASSIGN, R3                      ;
              52     0000'  CF  9E 00009           MOVAB   NON_AST_QUEUE, R2                   ;
              5E            04  C2 0000E           SUBL2   #4, SP                              ;
              01        04  BC  D1 00011           CMPL    @STRUCTURE_LEVEL, #1                ; 1298
                            08  13 00015           BEQL    1$                                  ;
              50 00000000G  8F  D0 00017           MOVL    #SMB$_INVSTRLEV, R0                 ; 1300
                                04 0001E           RET
                            7E  7C 0001F 1$:       CLRQ    -(SP)                               ; 1302
                        18  A2  9F 00021           PUSHAB  IMBX_CHAN                           ;
                        B1  AF  9F 00024           PUSHAB  P.AAA                               ;
              63            04  FB 00027           CALLS   #4, SYS$ASSIGN                      ;
              66            50  E9 0002A           BLBC    STATUS, 4$                          ;
                            7E  7C 0002D           CLRQ    -(SP)                               ; 1304
                        1A  A2  9F 0002F           PUSHAB  OMBX_CHAN                           ;
                        B7  AF  9F 00032           PUSHAB  P.AAC                               ;
              63            04  FB 00035           CALLS   #4, SYS$ASSIGN                      ;
              58            50  E9 00038           BLBC    STATUS, 4$                          ;
              62            62  9E 0003B           MOVAB   NON_AST_QUEUE, NON_AST_QUEUE        ; 1306
          04  A2            62  9E 0003E           MOVAB   NON_AST_QUEUE, NON_AST_QUEUE+4      ;
          08  A2        08  A2  9E 00042           MOVAB   MESSAGE_QUEUE, MESSAGE_QUEUE        ; 1307
          0C  A2        08  A2  9E 00047           MOVAB   MESSAGE_QUEUE, MESSAGE_QUEUE+4      ;
              02            6C  91 0004C           CMPB    (AP), #2                            ; 1309
```

```
                                        0A  1F  0004F             BLSSU   2$                                                      ;
                                    08  AC  D5  00051             TSTL    8(AP)                                                   ;
                                    05  13  00054                 BEQL    2$                                                      ;
                                10  A2      08  AC  D0  00056     MOVL    AST_ROUTINE, MESSAGE_AST_ROUTINE                        ; 1311
                                14  A2      01  D0  0005B 2$:    MOVL    #1, MAXIMUM_STREAMS                                     ; 1313
                                    03      6C  91  0005F         CMPB    (AP), #3                                                ; 1314
                                            0A  1F  00062         BLSSU   3$                                                      ;
                                    0C      AC  D5  00064         TSTL    12(AP)                                                  ;
                                            05  13  00067         BEQL    3$                                                      ;
                                14  A2      0C  BC  D0  00069     MOVL    @STREAMS, MAXIMUM_STREAMS                               ; 1316
                                        83  AF  9F  0006E 3$:     PUSHAB  P.AAE                                                   ; 1318
                        00000000G   00      01  FB  00071         CALLS   #1, SYS$PURGWS                                          ;
                                    18      50  E9  00078         BLBC    STATUS, 4$                                              ;
                                        FF7D CF  9F  0007B         PUSHAB  P.AAF                                                   ; 1320
                                        04  AE  9F  0007F         PUSHAB  IOB                                                     ;
                        00000000G   00      02  FB  00082         CALLS   #2, PSM$ALLOCATE_IOB                                    ;
                                            6E  DD  00089         PUSHL   IOB                                                     ; 1321
                            0000V   CF      01  FB  0008B         CALLS   #1, READ_MAILBOX                                        ;
                                    50      01  D0  00090         MOVL    #1, R0                                                  ; 1325
                                            04  00093 4$:         RET                                                             ;

; Routine Size:  148 bytes,    Routine Base:  CODE + 0100
```

```
;  385     1326  1 GLOBAL ROUTINE SMB$READ_MESSAGE ( %SBTTL 'READ_MESSAGE'
;  386     1327  1           STREAM  : REF $LONGWORD,
;  387     1328  1           BUFFER  ,
;  388     1329  1           REQUEST : REF $LONGWORD,
;  389     1330  1           SIZE    : REF $WORD             !*! NOT REFERENCED !
;  390     1331  1           ) =
;  391     1332  2 BEGIN
;  392     1333  2           PARAMETER_INDEX_ (STREAM, BUFFER, REQUEST, SIZE);
;  393     1334  2
;  394     1335  2 !++
;  395     1336  2 !
;  396     1337  2 ! FUNCTIONAL DESCRIPTION:
;  397     1338  2 !       Copies the next message into the caller's buffer.                                   ;
;  398     1339  2 !
;  399     1340  2 ! INPUT PARAMETERS:
;  400     1341  2 !       STREAM          - Address of longword to receive stream number.
;  401     1342  2 !       BUFFER          - Descriptor of output buffer.
;  402     1343  2 !       REQUEST         - Address of longword to receive message request code.
;  403     1344  2 !       SIZE            - Address of word to receive size of message.
;  404     1345  2 !
;  405     1346  2 ! IMPLICIT INPUTS:
;  406     1347  2 !       <tbs>           - Message queue header.
;  407     1348  2 !
;  408     1349  2 ! OUTPUT PARAMETERS:
;  409     1350  2 !       NONE
;  410     1351  2 !
;  411     1352  2 ! IMPLICIT OUTPUTS:
;  412     1353  2 !       NONE
;  413     1354  2 !
;  414     1355  2 ! ROUTINE VALUE:
;  415     1356  2 !       SS$_NORMAL      - Normal successful completion.
;  416     1357  2 !       LIB$_INVARG     - Invalid argument.
;  417     1358  2 !       ...             - Any return from LIB$SCOPY_xxxx routines.
;  418     1359  2 !
;  419     1360  2 ! SIDE EFFECTS:
;  420     1361  2 !       The message is dequeued from the message queue.  If no message
;  421     1362  2 !       is available the thread is blocked until one is received.
;  422     1363  2 !
;  423     1364  2 !--
;  424     1365  2
;  425     1366  2 LOCAL
;  426     1367  2           DSB:            REF $BBLOCK,
;  427     1368  2           MESSAGE:        REF $BBLOCK
;  428     1369  2           ;
;  429     1370  2
;  430     1371  2 ! Hibernate until there is work to do
;  431     1372  2 !
;  432     1373  2 WHILE REMOVE_HEAD_ (DSB, MESSAGE_QUEUE) DO $HIBER;
;  433     1374  2 DSB = .DSB - $BYTEOFFSET (DSB_Q_QLINKS);
;  434     1375  2
;  435     1376  2 MESSAGE = .DESC_ADDR_ (DSB[DSB_Q_DESC]);
;  436     1377  2
;  437     1378  2 REQUEST[] = .MESSAGE[SMBMSG$W_REQUEST_CODE];
;  438     1379  2 IF PARAMETER_PRESENT_ (SIZE)
;  439     1380  2 THEN
;  440     1381  2     $BBLOCK [.SIZE,0,0,16,0] = .DESC_SIZE_ (DSB[DSB_Q_DESC]);
;  441     1382  2
```

```
;  442          1383  2 STREAM[] = .MESSAGE[SMBMSG$B_STREAM_INDEX];
;  443          1384  2
;  444          1385  2 COPY_DX_DX_ (DSB[DSB_Q_DESC], .BUFFER);
;  445          1386  2
;  446          1387  2 PSM$DEALLOCATE_DSB (.DSB);
;  447          1388  2
;  448          1389  2 SS$_NORMAL
;  449          1390  2
;  450          1391  1 END;


                                                                .EXTRN  SYS$HIBER

                                                 000C 00000     .ENTRY  SMB$READ_MESSAGE, Save R2,R3                    ; 1326
                        52      0000'    DF  0F 00002 1$:     REMQUE  @MESSAGE_QUEUE, DSB                             ; 1373
                                         09  1C 00007          BVC     2$
                00000000G       00       00  FB 00009          CALLS   #0, SYS$HIBER
                                         F0  11 00010          BRB     1$
                        51        0C     A2  D0 00012 2$:     MOVL    12(DSB), MESSAGE                                ; 1376
                     0C BC                61  3C 00016          MOVZWL  (MESSAGE), @REQUEST                             ; 1378
                        04                6C  91 0001A          CMPB    (AP), #4                                        ; 1379
                                          0D  1F 0001D          BLSSU   3$
                              10          AC  D5 0001F          TSTL    16(AP)
                                          08  13 00022          BEQL    3$
                        50                52  D0 00024          MOVL    DSB, R0                                         ; 1381
                     10 BC          08    A0  B0 00027          MOVW    8(R0), @SIZE
                     04 BC          03    A1  9A 0002C 3$:     MOVZBL  3(MESSAGE), @STREAM                             ; 1383
                                    08    A2  9F 00031          PUSHAB  8(R2)                                           ; 1385
                                    08    AC  DD 00034          PUSHL   BUFFER
                00000000G       00          02  FB 00037          CALLS   #2, STR$COPY_DX
                                53          50  D0 0003E          MOVL    R0, STATUS
                                09          53  E8 00041          BLBS    STATUS, 4$
                                            53  DD 00044          PUSHL   STATUS
                00000000G       00          01  FB 00046          CALLS   #1, LIB$SIGNAL
                                            52  DD 0004D 4$:     PUSHL   DSB                                             ; 1387
                00000000G       00          01  FB 0004F          CALLS   #1, PSM$DEALLOCATE_DSB
                                50          01  D0 00056          MOVL    #1, R0                                          ; 1391
                                                04 00059          RET

; Routine Size:  90 bytes,    Routine Base:  CODE + 0194
```

```
;  452          1392  1 GLOBAL ROUTINE PSM$SCHEDULE_NON_AST ( %SBTTL 'SCHEDULE_NON_AST'
;  453          1393  1         PARAMETER
;  454          1394  1         ) =
;  455          1395  2 BEGIN
;  456          1396  2 !++
;  457          1397  2 !
;  458          1398  2 ! FUNCTIONAL DESCRIPTION:
;  459          1399  2 !       Schedules execution at non-ast level.
;  460          1400  2 !
;  461          1401  2 ! INPUT PARAMETERS:
;  462          1402  2 !       PARAMTER        - Descriptor of a paramter block to be queued to
;  463          1403  2 !                         non-ast level.
;  464          1404  2 !
;  465          1405  2 ! IMPLICIT INPUTS:
;  466          1406  2 !       <tbs>           - Work queue header.
;  467          1407  2 !       <tbs>           - Descriptor queue header.
;  468          1408  2 !
;  469          1409  2 ! OUTPUT PARAMETERS:
;  470          1410  2 !       NONE
;  471          1411  2 !
;  472          1412  2 ! IMPLICIT OUTPUTS:
;  473          1413  2 !       NONE
;  474          1414  2 !
;  475          1415  2 ! ROUTINE VALUE:
;  476          1416  2 !       SS$_NORMAL      - Normal, successful completion.
;  477          1417  2 !       ...             - Any return from LIB$SCOPY_xxxx routines.
;  478          1418  2 !
;  479          1419  2 ! SIDE EFFECTS:
;  480          1420  2 !       The parameter block is copied to an internal buffer and placed in
;  481          1421  2 !       the work queue.  A wakeup is scheduled if the queue was empty.
;  482          1422  2 !
;  483          1423  2 !--
;  484          1424  2
;  485          1425  2 LOCAL
;  486          1426  2         DSB : REF $BBLOCK
;  487          1427  2         ;
;  488          1428  2
;  489          1429  2 PSM$ALLOCATE_DSB (DSB);
;  490          1430  2
;  491          1431  2 COPY_DX_DX_ (.PARAMETER, DSB[DSB_Q_DESC]);
;  492          1432  2
;  493          1433  2 IF INSERT_TAIL_ (DSB[DSB_Q_QLINKS], NON_AST_QUEUE)      ! True if empty
;  494          1434  2 THEN
;  495          1435  2     SIGNAL_IF_ERROR_ ($WAKE ());
;  496          1436  2
;  497          1437  2 SS$_NORMAL
;  498          1438  2
;  499          1439  1 END;


                                                .EXTRN  SYS$WAKE

                                  000C 00000    .ENTRY  PSM$SCHEDULE_NON_AST, Save R2,R3        ; 1392
                 53 00000000G   00  9E 00002    MOVAB   LIB$SIGNAL, R3                          ;
                 5E             04  C2 00009    SUBL2   #4, SP                                  ;
                                5E  DD 0000C    PUSHL   SP                                      ; 1429
```

```
                00000000G  00            01  FB 0000E           CALLS   #1, PSM$ALLOCATE_DSB
                                     04   AC  DD 00015           PUSHL   PARAMETER                          ;  1431
                      7E        04   AE   08  C1 00018           ADDL3   #8, DSB, -(SP)
                00000000G  00            02  FB 0001D           CALLS   #2, STR$COPY_DX
                           52            50  D0 00024           MOVL    R0, STATUS
                           05            52  E8 00027           BLBS    STATUS, 1$
                                         52  DD 0002A           PUSHL   STATUS
                           63            01  FB 0002C           CALLS   #1, LIB$SIGNAL
                    0000'  DF        00   BE  0E 0002F 1$:      INSQUE  @DSB, @NON_AST_QUEUE+4             ;  1433
                                         14  12 00035           BNEQ    2$
                                         7E  7C 00037           CLRQ    -(SP)                              ;  1435
                00000000G  00            02  FB 00039           CALLS   #2, SYS$WAKE
                           52            50  D0 00040           MOVL    R0, STATUS
                           05            52  E8 00043           BLBS    STATUS, 2$
                                         52  DD 00046           PUSHL   STATUS
                           63            01  FB 00048           CALLS   #1, LIB$SIGNAL
                           50            01  D0 0004B 2$:       MOVL    #1, R0                             ;  1439
                                             04 0004E           RET

; Routine Size:  79 bytes,    Routine Base:  CODE + 01EE
```

```
;  501          1440  1 GLOBAL ROUTINE SMB$SEND_TO_JOBCTL ( %SBTTL 'SEND_RESPONSE'
;  502          1441  1         STREAM          : REF $LONGWORD,
;  503          1442  1         REQUEST         : REF $LONGWORD,
;  504          1443  1         ACCOUNTING,
;  505          1444  1         CHECKPOINT,
;  506          1445  1         DEVICE_STATUS   : REF $LONGWORD,
;  507          1446  1         ERROR           : REF VECTOR
;  508          1447  1         ) =
;  509          1448  2 BEGIN
;  510        P 1449  2         PARAMETER_INDEX  (STREAM, REQUEST, ACCOUNTING, CHECKPOINT,
;  511          1450  2             DEVICE_STATUS, ERROR);
;  512          1451  2
;  513          1452  2 !++
;  514          1453  2 !
;  515          1454  2 ! FUNCTIONAL DESCRIPTION:
;  516          1455  2 !       Sends a response to the controlling process
;  517          1456  2 !
;  518          1457  2 ! INPUT PARAMETERS:
;  519          1458  2 !       STREAM          - Stream number.
;  520          1459  2 !       REQUEST         - [optional] Request being responded to.
;  521          1460  2 !       ACCOUNTING      - [optional] Descriptor of accounting area.
;  522          1461  2 !       CHECKPOINT      - [optional] Descriptor of checkpoint area.
;  523          1462  2 !       DEVICE_STATUS   - [optional] Device status
;  524          1463  2 !       ERROR           - [optional] Counted vector of condition codes.
;  525          1464  2 !
;  526          1465  2 ! IMPLICIT INPUTS:
;  527          1466  2 !       NONE
;  528          1467  2 !
;  529          1468  2 ! OUTPUT PARAMETERS:
;  530          1469  2 !       NONE
;  531          1470  2 !
;  532          1471  2 ! IMPLICIT OUTPUTS:
;  533          1472  2 !       NONE
;  534          1473  2 !
;  535          1474  2 ! ROUTINE VALUE:
;  536          1475  2 !       SS$_NORMAL      - Normal, successful completion.
;  537          1476  2 !       LIB$_INVARG     - Invalid argument.
;  538          1477  2 !
;  539          1478  2 ! SIDE EFFECTS:
;  540          1479  2 !       A mailbox message is sent to the controlling process.
;  541          1480  2 !
;  542          1481  2 !--
;  543          1482  2
;  544          1483  2 LOCAL
;  545          1484  2         DSB             : REF $BBLOCK,
;  546          1485  2         MSG_SIZE,
;  547          1486  2         MSG_ADDR        : REF $BBLOCK,
;  548          1487  2         ACC_SIZE        : INITIAL (0),
;  549          1488  2         ACC_ADDR,
;  550          1489  2         CKP_SIZE        : INITIAL (0),
;  551          1490  2         CKP_ADDR,
;  552          1491  2         DEV_SIZE        : INITIAL (0),
;  553          1492  2         ERR_SIZE        : INITIAL (0),
;  554          1493  2         MAX_SIZE        : INITIAL (0),
;  555          1494  2         REQ_SIZE        : INITIAL (0)
;  556          1495  2         ;
;  557          1496  2
```

```
;  558     1497  2 ! Get size of request item
;  559     1498  2 !
;  560     1499  2 IF PARAMETER_PRESENT_ (REQUEST)
;  561     1500  2 THEN
;  562     1501  3     BEGIN
;  563     1502  3     REQ_SIZE = 4;
;  564     1503  3     IF .REQUEST[] EQL SMBMSG$K_START_STREAM
;  565     1504  3     THEN
;  566     1505  3         MAX_SIZE = 4;
;  567     1506  2     END;
;  568     1507  2
;  569     1508  2
;  570     1509  2 ! Get size of accounting data.
;  571     1510  2 !
;  572     1511  2 IF PARAMETER_PRESENT_ (ACCOUNTING)
;  573     1512  2 THEN
;  574     1513  2     STR$ANALYZE_SDESC (.ACCOUNTING, ACC_SIZE, ACC_ADDR);
;  575     1514  2
;  576     1515  2
;  577     1516  2 ! Get size of checkpoint data.
;  578     1517  2 !
;  579     1518  2 IF PARAMETER_PRESENT_ (CHECKPOINT)
;  580     1519  2 THEN
;  581     1520  2     STR$ANALYZE_SDESC (.CHECKPOINT, CKP_SIZE, CKP_ADDR);
;  582     1521  2
;  583     1522  2
;  584     1523  2 ! Get size of device status item
;  585     1524  2 !
;  586     1525  2 IF PARAMETER_PRESENT_ (DEVICE_STATUS)
;  587     1526  2 THEN
;  588     1527  2     DEV_SIZE = 4;
;  589     1528  2
;  590     1529  2
;  591     1530  2 ! Get size of error data
;  592     1531  2 !
;  593     1532  2 IF PARAMETER_PRESENT_ (ERROR)
;  594     1533  2 THEN
;  595     1534  2     ERR_SIZE = .ERROR[0] * 4;
;  596     1535  2
;  597     1536  2
;  598     1537  2 ! Compute maximum message size
;  599     1538  2 !
;  600     1539  2 MSG_SIZE =       SMBMSG$S_REQUEST_HEADER          ! - message header
;  601     1540  2
;  602     1541  2                 + SMBMSG$S_ITEM_HEADER           ! - request_response item header
;  603     1542  2                 + .REQ_SIZE                      ! - request_response size
;  604     1543  2
;  605     1544  2                 + SMBMSG$S_ITEM_HEADER           ! - accounting_data item header
;  606     1545  2                 + .ACC_SIZE                      ! - accounting_data size
;  607     1546  2
;  608     1547  2                 + SMBMSG$S_ITEM_HEADER           ! - checkpoint_data item header
;  609     1548  2                 + .CKP_SIZE                      ! - checkpoint_data size
;  610     1549  2
;  611     1550  2                 + SMBMSG$S_ITEM_HEADER           ! - device_status item header
;  612     1551  2                 + .DEV_SIZE                      ! - device_status size
;  613     1552  2
;  614     1553  2                 + SMBMSG$S_ITEM_HEADER           ! - condition vector item header
```

```
;  615          1554  2                 + .ERR_SIZE                     ! - condition vector size
;  616          1555  2
;  617          1556  2                 + SMBMSG$S_ITEM_HEADER          ! - maximum streams item header
;  618          1557  2                 + .MAX_SIZE                     ! - maximum streams size
;  619          1558  2
;  620          1559  2                 + 4;                            ! - trailing 0 longword
;  621          1560  2
;  622          1561  2
;  623          1562  2 ! Allocate a DSB and reserve sufficient space for message
;  624          1563  2 !
;  625          1564  2 PSM$ALLOCATE_DSB (DSB, MSG_SIZE);
;  626          1565  2 MSG_ADDR = .DESC_ADDR_ (DSB[DSB_Q_DESC]);
;  627          1566  2
;  628          1567  2
;  629          1568  2 ! Request header.
;  630          1569  2 !
;  631          1570  2 MSG_ADDR[SMBMSG$W_REQUEST_CODE] = MSG$_SMBINI;
;  632          1571  2 MSG_ADDR[SMBMSG$B_STRUCTURE_LEVEL] = SMBMSG$K_STRUCTURE_LEVEL;
;  633          1572  2 !*! MSG_ADDR[SMBMSG$B_STRUCTURE_LEVEL] = SMBMSG$K_LEVELT;
;  634          1573  2 MSG_ADDR[SMBMSG$B_STREAM_INDEX] = .STREAM[];
;  635          1574  2 MSG_ADDR = .MSG_ADDR + SMBMSG$S_REQUEST_HEADER;
;  636          1575  2
;  637          1576  2
;  638          1577  2 ! Request status item.
;  639          1578  2 !
;  640          1579  2 IF .REQ_SIZE NEQ 0
;  641          1580  2 THEN
;  642          1581  3     BEGIN
;  643          1582  3     MSG_ADDR[SMBMSG$W_ITEM_SIZE] = .REQ_SIZE;
;  644          1583  3     MSG_ADDR[SMBMSG$W_ITEM_CODE] = SMBMSG$K_REQUEST_RESPONSE;
;  645          1584  3     MSG_ADDR = .MSG_ADDR + SMBMSG$S_ITEM_HEADER;
;  646          1585  3     MSG_ADDR[PSM$L_] = .REQUEST[];
;  647          1586  3     MSG_ADDR = .MSG_ADDR + .REQ_SIZE;
;  648          1587  2     END;
;  649          1588  2
;  650          1589  2
;  651          1590  2 ! Accounting data item
;  652          1591  2 !
;  653          1592  2 IF .ACC_SIZE NEQ 0
;  654          1593  2 THEN
;  655          1594  3     BEGIN
;  656          1595  3     MSG_ADDR[SMBMSG$W_ITEM_SIZE] = .ACC_SIZE;
;  657          1596  3     MSG_ADDR[SMBMSG$W_ITEM_CODE] = SMBMSG$K_ACCOUNTING_DATA;
;  658          1597  3     MSG_ADDR = .MSG_ADDR + SMBMSG$S_ITEM_HEADER;
;  659          1598  3     MSG_ADDR = CH$MOVE (.ACC_SIZE, .ACC_ADDR, .MSG_ADDR);
;  660          1599  2     END;
;  661          1600  2
;  662          1601  2
;  663          1602  2 ! Checkpoint data item
;  664          1603  2 !
;  665          1604  2 IF .CKP_SIZE NEQ 0
;  666          1605  2 THEN
;  667          1606  3     BEGIN
;  668          1607  3     MSG_ADDR[SMBMSG$W_ITEM_SIZE] = .CKP_SIZE;
;  669          1608  3     MSG_ADDR[SMBMSG$W_ITEM_CODE] = SMBMSG$K_CHECKPOINT_DATA;
;  670          1609  3     MSG_ADDR = .MSG_ADDR + SMBMSG$S_ITEM_HEADER;
;  671          1610  3     MSG_ADDR = CH$MOVE (.CKP_SIZE, .CKP_ADDR, .MSG_ADDR);
```

```
;  672          1611  2     END;
;  673          1612  2
;  674          1613  2
;  675          1614  2 ! Device status item.
;  676          1615  2 !
;  677          1616  2 IF .DEV_SIZE NEQ 0
;  678          1617  2 THEN
;  679          1618  3     BEGIN
;  680          1619  3     MSG_ADDR[SMBMSG$W_ITEM_SIZE] = .DEV_SIZE;
;  681          1620  3     MSG_ADDR[SMBMSG$W_ITEM_CODE] = SMBMSG$K_DEVICE_STATUS;
;  682          1621  3     MSG_ADDR = .MSG_ADDR + SMBMSG$S_ITEM_HEADER;
;  683          1622  3     MSG_ADDR[PSM$L_] = .DEVICE_STATUS[];
;  684          1623  3     MSG_ADDR = .MSG_ADDR + .DEV_SIZE;
;  685          1624  2     END;
;  686          1625  2
;  687          1626  2
;  688          1627  2 ! Condition vector item.
;  689          1628  2 !
;  690          1629  2 IF .ERR_SIZE NEQ 0
;  691          1630  2 THEN
;  692          1631  3     BEGIN
;  693          1632  3     MSG_ADDR[SMBMSG$W_ITEM_SIZE] = .ERR_SIZE;
;  694          1633  3     MSG_ADDR[SMBMSG$W_ITEM_CODE] = SMBMSG$K_CONDITION_VECTOR;
;  695          1634  3     MSG_ADDR = .MSG_ADDR + SMBMSG$S_ITEM_HEADER;
;  696          1635  3     MSG_ADDR = CH$MOVE (.ERR_SIZE, ERROR[1], .MSG_ADDR);
;  697          1636  2     END;
;  698          1637  2
;  699          1638  2
;  700          1639  2 ! Maximum streams item.
;  701          1640  2 !
;  702          1641  2 IF .MAX_SIZE NEQ 0
;  703          1642  2 THEN
;  704          1643  3     BEGIN
;  705          1644  3     MSG_ADDR[SMBMSG$W_ITEM_SIZE] = .MAX_SIZE;
;  706          1645  3     MSG_ADDR[SMBMSG$W_ITEM_CODE] = SMBMSG$K_MAXIMUM_STREAMS;
;  707          1646  3     MSG_ADDR = .MSG_ADDR + SMBMSG$S_ITEM_HEADER;
;  708          1647  3     MSG_ADDR[PSM$L_] = .MAXIMUM_STREAMS;
;  709          1648  3     MSG_ADDR = .MSG_ADDR + .MAX_SIZE;
;  710          1649  2     END;
;  711          1650  2
;  712          1651  2
;  713          1652  2 ! Trailing zero longword
;  714          1653  2 !
;  715          1654  2 .MSG_ADDR = 0;
;  716          1655  2 MSG_ADDR = .MSG_ADDR + 4;
;  717          1656  2
;  718          1657  2
;  719          1658  2 WRITE_MAILBOX (.DSB, .MSG_ADDR - .DESC_ADDR_ (DSB[DSB_Q_DESC]));
;  720          1659  2
;  721          1660  2 SS$_NORMAL
;  722          1661  2
;  723          1662  1 END;
```

```
                         07FC 00000          .ENTRY  SMB$SEND_TO_JOBCTL, Save R2,R3,R4,R5,R6,R7,-; 1440
                                                     R8,R9,R10
                5A 00000000G   00    9E 00002       MOVAB   STR$ANALYZE_SDESC, R10                 :
                5E                   18    C2 00009       SUBL2   #24, SP                            :
                         04    AE    D4 0000C       CLRL    ACC_SIZE                               ; 1448
                         0C    AE    D4 0000F       CLRL    CKP_SIZE                               :
                               58    7C 00012       CLRQ    ERR_SIZE                               :
                               57    D4 00014       CLRL    MAX_SIZE                               :
                               52    D4 00016       CLRL    REQ_SIZE                               :
                02             6C    91 00018       CMPB    (AP), #2                               ; 1499
                               11    1F 0001B       BLSSU   1$                                     :
                         08    AC    D5 0001D       TSTL    8(AP)                                  :
                               0C    13 00020       BEQL    1$                                     :
                52             04    D0 00022       MOVL    #4, REQ_SIZE                           ; 1502
                04       08    BC    D1 00025       CMPL    @REQUEST, #4                           ; 1503
                               03    12 00029       BNEQ    1$                                     :
                57             04    D0 0002B       MOVL    #4, MAX_SIZE                           ; 1505
                03             6C    91 0002E 1$:   CMPB    (AP), #3                               ; 1511
                               10    1F 00031       BLSSU   2$                                     :
                         0C    AC    D5 00033       TSTL    12(AP)                                 :
                               0B    13 00036       BEQL    2$                                     :
                               5E    DD 00038       PUSHL   SP                                     ; 1513
                         08    AE    9F 0003A       PUSHAB  ACC_SIZE                               :
                         0C    AC    DD 0003D       PUSHL   ACCOUNTING                             :
                6A             03    FB 00040       CALLS   #3, STR$ANALYZE_SDESC                  :
                04             6C    91 00043 2$:   CMPB    (AP), #4                               ; 1518
                               11    1F 00046       BLSSU   3$                                     :
                         10    AC    D5 00048       TSTL    16(AP)                                 :
                               0C    13 0004B       BEQL    3$                                     :
                         08    AE    9F 0004D       PUSHAB  CKP_ADDR                               ; 1520
                         10    AE    9F 00050       PUSHAB  CKP_SIZE                               :
                         10    AC    DD 00053       PUSHL   CHECKPOINT                             :
                6A             03    FB 00056       CALLS   #3, STR$ANALYZE_SDESC                  :
                05             6C    91 00059 3$:   CMPB    (AP), #5                               ; 1525
                               08    1F 0005C       BLSSU   4$                                     :
                         14    AC    D5 0005E       TSTL    20(AP)                                 :
                               03    13 00061       BEQL    4$                                     :
                59             04    D0 00063       MOVL    #4, DEV_SIZE                           ; 1527
                06             6C    91 00066 4$:   CMPB    (AP), #6                               ; 1532
                               0A    1F 00069       BLSSU   5$                                     :
                         18    AC    D5 0006B       TSTL    24(AP)                                 :
                               05    13 0006E       BEQL    5$                                     :
          58         18    BC       02    78 00070       ASHL    #2, @ERROR, ERR_SIZE               ; 1534
          50             52    04    AE    C1 00075 5$:   ADDL3   ACC_SIZE, REQ_SIZE, R0          ; 1545
                         50    0C    AE    C0 0007A       ADDL2   CKP_SIZE, R0                   ; 1548
                         50          59    C0 0007E       ADDL2   DEV_SIZE, R0                   ; 1551
                         50          58    C0 00081       ADDL2   ERR_SIZE, R0                   ; 1554
                   10    AE       20 A740    9E 00084       MOVAB   32(MAX_SIZE)[R0], MSG_SIZE     ; 1559
                         10    AE    9F 0008A       PUSHAB  MSG_SIZE                               ; 1564
                         18    AE    9F 0008D       PUSHAB  DSB                                    :
     00000000G           00          02    FB 00090       CALLS   #2, PSM$ALLOCATE_DSB           :
                         56    14    AE    D0 00097       MOVL    DSB, R6                        ; 1565
                         53    0C    A6    D0 0009B       MOVL    12(R6), MSG_ADDR               :
                         83          08    B0 0009F       MOVW    #8, (MSG_ADDR)+                ; 1570
                         83          01    90 000A2       MOVB    #1, (MSG_ADDR)+                ; 1571
                         83    04    BC    90 000A5       MOVB    @STREAM, (MSG_ADDR)+           ; 1573
                               52          D5 000A9       TSTL    REQ_SIZE                       ; 1579
```

```
                                 0D  13 000AB          BEQL    6$                                      ;
                           83    52  B0 000AD          MOVW    REQ_SIZE, (MSG_ADDR)+                   ; 1582
                           83    30  B0 000B0          MOVW    #48, (MSG_ADDR)+                        ; 1583
                           63 08 BC  D0 000B3          MOVL    @REQUEST, (MSG_ADDR)                    ; 1585
                           53    52  C0 000B7          ADDL2   REQ_SIZE, MSG_ADDR                      ; 1586
                        04 AE    D5 000BA 6$:          TSTL    ACC_SIZE                                ; 1592
                                 0D  13 000BD          BEQL    7$                                      ;
                           83 04 AE  B0 000BF          MOVW    ACC_SIZE, (MSG_ADDR)+                   ; 1595
                           83    01  B0 000C3          MOVW    #1, (MSG_ADDR)+                         ; 1596
                  63 00 BE 04 AE  28 000C6          MOVC3   ACC_SIZE, @ACC_ADDR, (MSG_ADDR)         ; 1598
                        0C AE    D5 000CC 7$:          TSTL    CKP_SIZE                                ; 1604
                                 0D  13 000CF          BEQL    8$                                      ;
                           83 0C AE  B0 000D1          MOVW    CKP_SIZE, (MSG_ADDR)+                   ; 1607
                           83    07  B0 000D5          MOVW    #7, (MSG_ADDR)+                         ; 1608
                  63 08 BE 0C AE  28 000D8          MOVC3   CKP_SIZE, @CKP_ADDR, (MSG_ADDR)         ; 1610
                                 59  D5 000DE 8$:      TSTL    DEV_SIZE                                ; 1616
                                 0D  13 000E0          BEQL    9$                                      ;
                           83    59  B0 000E2          MOVW    DEV_SIZE, (MSG_ADDR)+                   ; 1619
                           83    0A  B0 000E5          MOVW    #10, (MSG_ADDR)+                        ; 1620
                           63 14 BC  D0 000E8          MOVL    @DEVICE_STATUS, (MSG_ADDR)              ; 1622
                           53    59  C0 000EC          ADDL2   DEV_SIZE, MSG_ADDR                      ; 1623
                                 58  D5 000EF 9$:      TSTL    ERR_SIZE                                ; 1629
                                 0F  13 000F1          BEQL    10$                                     ;
                           83    58  B0 000F3          MOVW    ERR_SIZE, (MSG_ADDR)+                   ; 1632
                           83    08  B0 000F6          MOVW    #8, (MSG_ADDR)+                         ; 1633
                           50 18 AC  D0 000F9          MOVL    ERROR, R0                               ; 1635
                  63 04 A0          58  28 000FD          MOVC3   ERR_SIZE, 4(R0), (MSG_ADDR)             ;
                                 57  D5 00102 10$:      TSTL    MAX_SIZE                                ; 1641
                                 0E  13 00104          BEQL    11$                                     ;
                           83    57  B0 00106          MOVW    MAX_SIZE, (MSG_ADDR)+                   ; 1644
                           83    1E  B0 00109          MOVW    #30, (MSG_ADDR)+                        ; 1645
                           63 0000' CF  D0 0010C          MOVL    MAXIMUM_STREAMS, (MSG_ADDR)             ; 1647
                           53    57  C0 00111          ADDL2   MAX_SIZE, MSG_ADDR                      ; 1648
                                 83  D4 00114 11$:      CLRL    (MSG_ADDR)+                             ; 1654
                  7E       53 0C A6  C3 00116          SUBL3   12(R6), MSG_ADDR, -(SP)                 ; 1658
                        18 AE    DD 0011B          PUSHL   DSB                                     ;
                0000V CF          02  FB 0011E          CALLS   #2, WRITE_MAILBOX                       ;
                           50     01  D0 00123          MOVL    #1, R0                                  ; 1662
                                 04  00126          RET                                             ;

; Routine Size:  295 bytes,    Routine Base:  CODE + 023D
```

```
;  725     1663  1 GLOBAL ROUTINE PSM$WAIT_FOR_NON_AST ( %SBTTL 'WAIT_FOR_NON_AST'
;  726     1664  1         BUFFER,
;  727     1665  1         SIZE    : REF $WORD                     !*! NOT REFERENCED ???
;  728     1666  1         ) =
;  729     1667  2 BEGIN
;  730     1668  2 !++
;  731     1669  2 !
;  732     1670  2 ! FUNCTIONAL DESCRIPTION:
;  733     1671  2 !       Waits for work scheduled by an ast-level routine.
;  734     1672  2 !
;  735     1673  2 ! INPUT PARAMETERS:
;  736     1674  2 !       BUFFER          - Descriptor of a buffer to receive the paramter block.
;  737     1675  2 !       SIZE            - Word to receive the paramter block size.
;  738     1676  2 !
;  739     1677  2 ! IMPLICIT INPUTS:
;  740     1678  2 !       <tbs>           - Work queue header.
;  741     1679  2 !       <tbs>           - Descriptor queue header.
;  742     1680  2 !
;  743     1681  2 ! OUTPUT PARAMETERS:
;  744     1682  2 !       NONE
;  745     1683  2 !
;  746     1684  2 ! IMPLICIT OUTPUTS:
;  747     1685  2 !       NONE
;  748     1686  2 !
;  749     1687  2 ! ROUTINE VALUE:
;  750     1688  2 !       SS$_NORMAL      - Normal successful completion.
;  751     1689  2 !       LIB$_INVARG     - Invalid argument.
;  752     1690  2 !       ...             - Any return from LIB$SCOPY_xxxx routines.
;  753     1691  2 !
;  754     1692  2 ! SIDE EFFECTS:
;  755     1693  2 !       NONE
;  756     1694  2 !
;  757     1695  2 !--
;  758     1696  2
;  759     1697  2 LOCAL
;  760     1698  2         DSB : REF $BBLOCK
;  761     1699  2         ;
;  762     1700  2
;  763     1701  2 ! Hibernate until there is work to do
;  764     1702  2 !
;  765     1703  2 WHILE REMOVE_HEAD_ (DSB, NON_AST_QUEUE) DO  $HIBER;
;  766     1704  2 DSB = .DSB - $BYTEOFFSET (DSB_Q_QLINKS);
;  767     1705  2
;  768     1706  2 COPY_DX_DX_ (DSB[DSB_Q_DESC], .BUFFER);
;  769     1707  2
;  770     1708  2 PSM$DEALLOCATE_DSB (.DSB);
;  771     1709  2
;  772     1710  2 SS$_NORMAL
;  773     1711  2
;  774     1712  1 END;


                                000C 00000          .ENTRY  PSM$WAIT_FOR_NON_AST, Save R2,R3        ; 1663
                    53  0000'   DF   0F 00002 1$:   REMQUE  @NON_AST_QUEUE, DSB                     ; 1703
```

```
                                    09  1C 00007          BVC     2$                                          ;
                 00000000G  00      00  FB 00009          CALLS   #0, SYS$HIBER                               ;
                                    F0  11 00010          BRB     1$                                          ;
                                08  A3  9F 00012 2$:      PUSHAB  8(DSB)                                      ; 1706
                                04  AC  DD 00015          PUSHL   BUFFER                                      ;
                 00000000G  00      02  FB 00018          CALLS   #2, STR$COPY_DX                             ;
                            52      50  D0 0001F          MOVL    R0, STATUS                                  ;
                            09      52  E8 00022          BLBS    STATUS, 3$                                  ;
                                    52  DD 00025          PUSHL   STATUS                                      ;
                 00000000G  00      01  FB 00027          CALLS   #1, LIB$SIGNAL                              ;
                                    53  DD 0002E 3$:      PUSHL   DSB                                         ; 1708
                 00000000G  00      01  FB 00030          CALLS   #1, PSM$DEALLOCATE_DSB                      ;
                            50      01  D0 00037          MOVL    #1, R0                                      ; 1712
                                        04 0003A          RET                                                 ;

; Routine Size:  59 bytes,    Routine Base:  CODE + 0364
```

```
;  776          1713  1 ROUTINE READ_MAILBOX ( %SBTTL 'READ_MAILBOX'
;  777          1714  1          IOB     : REF $BBLOCK
;  778          1715  1          ) : NOVALUE =
;  779          1716  2 BEGIN
;  780          1717  2 !++
;  781          1718  2 !
;  782          1719  2 ! FUNCTIONAL DESCRIPTION:
;  783          1720  2 !       Queues an ast read to the input mailbox.
;  784          1721  2 !
;  785          1722  2 ! INPUT PARAMETERS:
;  786          1723  2 !       IOB             - Address of an Input/Output control block
;  787          1724  2 !
;  788          1725  2 ! IMPLICIT INPUTS:
;  789          1726  2 !       IMBX_CHAN       - Input mailbox channel.
;  790          1727  2 !
;  791          1728  2 ! OUTPUT PARAMETERS:
;  792          1729  2 !       NONE
;  793          1730  2 !
;  794          1731  2 ! IMPLICIT OUTPUTS:
;  795          1732  2 !       NONE
;  796          1733  2 !
;  797          1734  2 ! ROUTINE VALUE:
;  798          1735  2 !       SS$_NORMAL      - Normal successful completion.
;  799          1736  2 !       ...             - Any QIO system service error.
;  800          1737  2 !
;  801          1738  2 ! SIDE EFFECTS:
;  802          1739  2 !       An read I/O is queued.
;  803          1740  2 !
;  804          1741  2 !--
;  805          1742  2
;  806        P 1743  2 SIGNAL_IF_ERROR_ ($QIO (
;  807        P 1744  2     FUNC=IO$_READVBLK,
;  808        P 1745  2     CHAN=.IMBX_CHAN,
;  809        P 1746  2     IOSB=IOB[IOB_Q_IOSB],
;  810        P 1747  2     ASTADR=READ_MAILBOX_AST,
;  811        P 1748  2     ASTPRM=.IOB,
;  812        P 1749  2     P1=.DESC_ADDR_ (IOB[IOB_Q_BUFFER]),
;  813        P 1750  2     P2=.DESC_SIZE_ (IOB[IOB_Q_BUFFER])
;  814          1751  2         ));
;  815          1752  2
;  816          1753  2 SS$_NORMAL
;  817          1754  2
;  818          1755  1 END;


                                                .EXTRN  SYS$QIO

                                0004 00000 READ_MAILBOX:
                                                .WORD   Save R2                         ; 1713
                                7E  7C 00002            CLRQ    -(SP)                   ; 1751
                                7E  7C 00004            CLRQ    -(SP)                   ;
                          51    04 AC  D0 00006         MOVL    IOB, R1                 ;
                          50    1C A1  9E 0000A         MOVAB   28(R1), R0              ;
                          7E       60  3C 0000E         MOVZWL  (R0), -(SP)             ;
                                04 A0  DD 00011         PUSHL   4(R0)                   ;
                                51 DD 00014             PUSHL   R1                      ;
```

```
                                 0000V  CF  9F 00016          PUSHAB  READ_MAILBOX_AST                ;
                                   0C   A1  9F 0001A          PUSHAB  12(RT)                          ;
                                        31  DD 0001D          PUSHL   #49                             ;
                            7E   0000'  CF  3C 0001F          MOVZWL  IMBX_CHAN, -(SP)                ;
                                        7E  D4 00024          CLRL    -(SP)                           ;
                 00000000G  00          0C  FB 00026          CALLS   #12, SYS$QIO                    ;
                            52          50  D0 0002D          MOVL    R0, STATUS                      ;
                            09          52  E8 00030          BLBS    STATUS, 1$                      ;
                                        52  DD 00033          PUSHL   STATUS                          ;
                 00000000G  00          01  FB 00035          CALLS   #1, LIB$SIGNAL                  ;
                                        04  0003C 1$:        RET                                     ; 1755

; Routine Size:  61 bytes,    Routine Base:  CODE + 039F
```

```
;  820     1756  1 ROUTINE READ_MAILBOX_AST ( %SBTTL 'READ_MAILBOX_AST'
;  821     1757  1         IOB     : REF $BBLOCK
;  822     1758  1         ) : NOVALUE =
;  823     1759  2 BEGIN
;  824     1760  2 !++
;  825     1761  2 !
;  826     1762  2 ! FUNCTIONAL DESCRIPTION:
;  827     1763  2 !       Reads a message from the input mailbox and performs basic
;  828     1764  2 !       validity checking.
;  829     1765  2 !
;  830     1766  2 ! INPUT PARAMETERS:
;  831     1767  2 !       IOB             - IO area structure, includes IOSB and buffer.
;  832     1768  2 !
;  833     1769  2 ! IMPLICIT INPUTS:
;  834     1770  2 !       NONE
;  835     1771  2 !
;  836     1772  2 ! OUTPUT PARAMETERS:
;  837     1773  2 !       NONE
;  838     1774  2 !
;  839     1775  2 ! IMPLICIT OUTPUTS:
;  840     1776  2 !       NONE
;  841     1777  2 !
;  842     1778  2 ! ROUTINE VALUE:
;  843     1779  2 !       NONE
;  844     1780  2 !
;  845     1781  2 ! SIDE EFFECTS:
;  846     1782  2 !       The message is placed in the message queue.  If user level is
;  847     1783  2 !       waiting for a message a wakeup is scheduled.
;  848     1784  2 !
;  849     1785  2 !--
;  850     1786  2
;  851     1787  2 LOCAL
;  852     1788  2         DSB:            REF $BBLOCK
;  853     1789  2         ;
;  854     1790  2
;  855     1791  2 BIND
;  856     1792  2         BUFFER = .DESC_ADDR_ (IOB[IOB_Q_BUFFER]) : $BBLOCK
;  857     1793  2         ;
;  858     1794  2
;  859     1795  2 ! Abort if any errors
;  860     1796  2 !
;  861     1797  2 IF NOT .IOB[IOB_W_IO_STATUS]
;  862     1798  2 THEN
;  863     1799  2     SIGNAL (.IOB[IOB_W_IO_STATUS]);
;  864     1800  2
;  865     1801  2 IF .BUFFER[SMBMSG$B_STREAM_INDEX] GEQU .MAXIMUM_STREAMS
;  866     1802  2 THEN
;  867     1803  3     BEGIN
;  868     1804  3     SMB$SEND_TO_JOBCTL (
;  869     1805  3         %REF (.BUFFER[SMBMSG$B_STREAM_INDEX]),  ! Stream number
;  870     1806  3         %REF (.BUFFER[SMBMSG$W_REQUEST_CODE]),  ! Responding to ...
;  871     1807  3         0,                                      ! No accounting
;  872     1808  3         0,                                      ! No checkpoint
;  873     1809  3         0,                                      ! No device status
;  874     1810  3         UPLIT (1, SMB$_INVSTMNBR)               ! Condition vector
;  875     1811  3         );
;  876     1812  3     READ_MAILBOX (.IOB);
```

```
;  877          1813  3     RETURN;
;  878          1814  2     END;
;  879          1815  2
;  880          1816  2
;  881          1817  2 ! Copy the message to dynamic memory
;  882          1818  2 !
;  883          1819  2 PSM$ALLOCATE_DSB (DSB);
;  884        P 1820  2 COPY_R_DX_ (IOB[IOB_W_IO_LENGTH], .DESC_ADDR_ (IOB[IOB_Q_BUFFER]),
;  885          1821  2                 DSB[DSB_Q_DESC]);
;  886          1822  2
;  887          1823  2
;  888          1824  2 ! Place in message queue and wake up user if empty
;  889          1825  2 !
;  890          1826  2 IF INSERT_TAIL_ (DSB[DSB_Q_QLINKS], MESSAGE_QUEUE)      ! True if empty
;  891          1827  2 THEN
;  892          1828  2     SIGNAL_IF_ERROR_ ($WAKE ());
;  893          1829  2
;  894          1830  2
;  895          1831  2 ! Queue another read from the mailbox.
;  896          1832  2 !
;  897          1833  2 READ_MAILBOX (.IOB);
;  898          1834  2
;  899          1835  2
;  900          1836  2 ! Call user AST routine if any
;  901          1837  2 !
;  902          1838  2 IF .MESSAGE_AST_ROUTINE NEQ 0
;  903          1839  2 THEN
;  904          1840  2     BLISS (.MESSAGE_AST_ROUTINE);
;  905          1841  2
;  906          1842  1 END;


                                        00000001  003DC P.AAG:   .LONG   1                                    ;
                                        00000000G 003E0         .LONG   SMB$_INVSTMNBR                       ;


                                                  003C 00000 READ_MAILBOX_AST:
                                                             .WORD   Save R2,R3,R4,R5                        ; 1756
                                    55        B6    AF  9E 00002         MOVAB   READ_MAILBOX, R5             ;
                                    54 00000000G    00  9E 00006         MOVAB   LIB$SIGNAL, R4               ;
                                    5E              0C  C2 0000D         SUBL2   #12, SP                      ;
                                    52        04    AC  D0 00010         MOVL    IOB, R2                      ; 1792
                                    53        20    A2  D0 00014         MOVL    32(R2), R3                   ;
                                    07        0C    A2  E8 00018         BLBS    12(R2), 1$                   ; 1797
                                    7E        0C    A2  3C 0001C         MOVZWL  12(R2), -(SP)                ; 1799
                                    64              01  FB 00020         CALLS   #1, LIB$SIGNAL               ;
0000'  CF        03    A3           08              00  ED 00023 1$:      CMPZV   #0, #8, 3(R3), MAXIMUM_STREAMS ; 1801
                                                    21  1F 0002B         BLSSU   2$                           ;
                                              C8    AF  9F 0002D         PUSHAB  P.AAG                        ; 1810
                                                    7E  7C 00030         CLRQ    -(SP)                        ; 1804
                                                    7E  D4 00032         CLRL    -(SP)                        ;
                                    14    AE              63  3C 00034         MOVZWL  (R3), 20(SP)         ; 1806
                                                    14    AE  9F 00038         PUSHAB  20(SP)               ;
                                    14    AE        03    A3  9A 0003B         MOVZBL  3(R3), 20(SP)        ; 1805
                                                    14    AE  9F 00040         PUSHAB  20(SP)               ;
```

```
                    FE9E   C5              06  FB 00043          CALLS   #6, SMB$SEND_TO_JOBCTL
                                            52  DD 00048          PUSHL   R2                                  ;1812
                           65               01  FB 0004A          CALLS   #1, READ_MAILBOX
                                            04  0004D            RET                                         ;1803
                                        08  AE  9F 0004E 2$:      PUSHAB  DSB                                 ;1819
                00000000G  00               01  FB 00051          CALLS   #1, PSM$ALLOCATE_DSB
                                        20  A2  DD 00058          PUSHL   32(R2)                              ;1821
                                        0E  A2  9F 0005B          PUSHAB  14(R2)
                        7E      10  AE          08  C1 0005E      ADDL3   #8, DSB, -(SP)
                00000000G  00               03  FB 00063          CALLS   #3, STR$COPY_R
                           52               50  D0 0006A          MOVL    R0, STATUS
                           05               52  E8 0006D          BLBS    STATUS, 3$
                                            52  DD 00070          PUSHL   STATUS
                           64               01  FB 00072          CALLS   #1, LIB$SIGNAL
                    0000'  DF           08  BE  0E 00075 3$:      INSQUE  @DSB, @MESSAGE_QUEUE+4              ;1826
                                            14  12 0007B          BNEQ    4$
                                            7E  7C 0007D          CLRQ    -(SP)                               ;1828
                00000000G  00               02  FB 0007F          CALLS   #2, SYS$WAKE
                           52               50  D0 00086          MOVL    R0, STATUS
                           05               52  E8 00089          BLBS    STATUS, 4$
                                            52  DD 0008C          PUSHL   STATUS
                           64               01  FB 0008E          CALLS   #1, LIB$SIGNAL
                                        04  AC  DD 00091 4$:      PUSHL   IOB                                 ;1833
                           65               01  FB 00094          CALLS   #1, READ_MAILBOX
                           50      0000'    CF  D0 00097          MOVL    MESSAGE_AST_ROUTINE, R0             ;1838
                                            03  13 0009C          BEQL    5$
                           60               00  FB 0009E          CALLS   #0, (R0)                            ;1840
                                            04  000A1 5$:        RET                                         ;1842

; Routine Size:  162 bytes,    Routine Base:  CODE + 03E4
```

```
;  908          1843  1 ROUTINE WRITE_MAILBOX ( %SBTTL 'WRITE_MAILBOX'
;  909          1844  1           DSB     : REF $BBLOCK,
;  910          1845  1           SIZE
;  911          1846  1           ) : NOVALUE =
;  912          1847  2 BEGIN
;  913          1848  2 !++
;  914          1849  2 !
;  915          1850  2 ! FUNCTIONAL DESCRIPTION:
;  916          1851  2 !       Queues a message to the output mailbox.
;  917          1852  2 !
;  918          1853  2 ! INPUT PARAMETERS:
;  919          1854  2 !       DSB             - Dynamic string block containing the message.
;  920          1855  2 !       SIZE            - True size of message (DSB gives buffer size)
;  921          1856  2 !
;  922          1857  2 ! IMPLICIT INPUTS:
;  923          1858  2 !       OMBX_CHAN       - Onput mailbox channel.
;  924          1859  2 !
;  925          1860  2 ! OUTPUT PARAMETERS:
;  926          1861  2 !       NONE
;  927          1862  2 !
;  928          1863  2 ! IMPLICIT OUTPUTS:
;  929          1864  2 !       NONE
;  930          1865  2 !
;  931          1866  2 ! ROUTINE VALUE:
;  932          1867  2 !       NONE
;  933          1868  2 !
;  934          1869  2 ! SIDE EFFECTS:
;  935          1870  2 !       A write I/O is queued.
;  936          1871  2 !
;  937          1872  2 !--
;  938          1873  2
;  939          1874  2
;  940        P 1875  2 SIGNAL_IF_ERROR_ ($QIO (
;  941        P 1876  2     FUNC=IO$_WRITEVBLK,
;  942        P 1877  2     CHAN=.OMBX_CHAN,
;  943        P 1878  2     P1=.DESC_ADDR_ (DSB[DSB_Q_DESC]),
;  944          1879  2     P2=.SIZE));
;  945          1880  2
;  946          1881  1 END;


                                        0004 00000 WRITE_MAILBOX:
                                                   .WORD   Save R2                                  ; 1843
                                        7E  7C 00002           CLRQ    -(SP)                        ; 1879
                                        7E  7C 00004           CLRQ    -(SP)
                                     08 AC  DD 00006           PUSHL   SIZE
                                  50 04 AC  D0 00009           MOVL    DSB, R0
                                     0C AC  DD 0000D           PUSHL   12(R0)
                                        7E  7C 00010           CLRQ    -(SP)
                                  7E    30  7D 00012           MOVQ    #48, -(SP)
                                  7E 0000' CF  3C 00015        MOVZWL  OMBX_CHAN, -(SP)
                                        7E  D4 0001A           CLRL    -(SP)
                           00000000G  00    0C  FB 0001C        CALLS   #12, SYS$QIO
                                  52        50  D0 00023        MOVL    R0, STATUS
```

```
                          09        52  E8 00026          BLBS    STATUS, 1$                    ;
                                    52  DD 00029          PUSHL   STATUS                        ;
                00000000G 00        01  FB 0002B          CALLS   #1, LIB$SIGNAL                ;
                                        04 00032 1$:      RET                                   ; 1881

; Routine Size:  51 bytes,    Routine Base:  CODE + 0486
```

```
;  948          1882  1 END
;  949          1883  0 ELUDOM


                                                        .EXTRN  LIB$SIGNAL, LIB$STOP

;                               PSECT SUMMARY

;       Name                    Bytes                   Attributes

;  DATA                           36  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  CODE                         1209  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)


;                             Library Statistics

;                                      -------- Symbols --------      Pages       Processing
;       File                           Total    Loaded   Percent      Mapped      Time

;  _$255$DUA28:[SYSLIB]LIB.L32;1       18619        49         0      1000          00:01.9


;                                   COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:SERVICES/OBJ=OBJ$:SERVICES MSRC$:SERVICES/UPDATE=(ENH$:SERVICES)

; Size:         1151 code + 94 data bytes
; Run Time:       00:30.8
; Elapsed Time:   00:38.5
; Lines/CPU Min:   3664
; Lexemes/CPU-Min: 28014
; Memory Used:  255 pages
; Compilation Complete
```

SERVICES
LIS

SMBMSG
LIS

TABLES
LIS