```
PPPPPPPPPPPP        AAAAAAAAA        SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
PPPPPPPPPPPP        AAAAAAAAA        SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
PPPPPPPPPPPP        AAAAAAAAA        SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
PPP          PPP   AAA         AAA  SSS          RRR         RRR        TTT        LLL
PPP          PPP   AAA         AAA  SSS          RRR         RRR        TTT        LLL
PPP          PPP   AAA         AAA  SSS          RRR         RRR        TTT        LLL
PPP          PPP   AAA         AAA  SSS          RRR         RRR        TTT        LLL
PPP          PPP   AAA         AAA  SSS          RRR         RRR        TTT        LLL
PPP          PPP   AAA         AAA  SSS          RRR         RRR        TTT        LLL
PPPPPPPPPPPP       AAA         AAA     SSSSSSSSS     RRRRRRRRRRR        TTT        LLL
PPPPPPPPPPPP       AAA         AAA     SSSSSSSSS     RRRRRRRRRRR        TTT        LLL
PPPPPPPPPPPP       AAA         AAA     SSSSSSSSS     RRRRRRRRRRR        TTT        LLL
PPP                AAAAAAAAAAAAAAA           SSS   RRR   RRR          TTT        LLL
PPP                AAAAAAAAAAAAAAA           SSS   RRR   RRR          TTT        LLL
PPP                AAAAAAAAAAAAAAA           SSS   RRR   RRR          TTT        LLL
PPP                AAA         AAA           SSS   RRR      RRR       TTT        LLL
PPP                AAA         AAA           SSS   RRR      RRR       TTT        LLL
PPP                AAA         AAA           SSS   RRR      RRR       TTT        LLL
PPP                AAA         AAA  SSSSSSSSSSS   RRR         RRR     TTT        LLLLLLLLLLLLLLL
PPP                AAA         AAA  SSSSSSSSSSS   RRR         RRR     TTT        LLLLLLLLLLLLLLL
PPP                AAA         AAA  SSSSSSSSSSS   RRR         RRR     TTT        LLLLLLLLLLLLLLL
PPP                AAA         AAA  SSSSSSSSSSS   RRR         RRR     TTT        LLLLLLLLLLLLLLL
```

```
PPPPPPPP      AAAAAA      SSSSSSSS    IIIIII      000000    HH      HH    AAAAAA    NN      NN  DDDDDDDD
PPPPPPPP      AAAAAA      SSSSSSSS    IIIIII      000000    HH      HH    AAAAAA    NN      NN  DDDDDDDD
PP      PP   AA      AA  SS            II      00      00  HH      HH   AA      AA   NN      NN  DD      DD
PP      PP   AA      AA  SS            II      00      00  HH      HH   AA      AA   NN      NN  DD      DD
PP      PP   AA      AA  SS            II      00      00  HH      HH   AA      AA   NNNN    NN  DD      DD
PP      PP   AA      AA  SS            II      00      00  HH      HH   AA      AA   NNNN    NN  DD      DD
PPPPPPPP     AA      AA    SSSSSS      II      00      00  HHHHHHHHHH   AA      AA   NN  NN  NN  DD      DD
PPPPPPPP     AA      AA    SSSSSS      II      00      00  HHHHHHHHHH   AA      AA   NN  NN  NN  DD      DD
PP           AAAAAAAAAA          SS    II      00      00  HH      HH   AAAAAAAAAA   NN    NNNN  DD      DD
PP           AAAAAAAAAA          SS    II      00      00  HH      HH   AAAAAAAAAA   NN    NNNN  DD      DD
PP           AA      AA          SS    II      00      00  HH      HH   AA      AA   NN      NN  DD      DD
PP           AA      AA          SS    II      00      00  HH      HH   AA      AA   NN      NN  DD      DD
PP           AA      AA  SSSSSSSS    IIIIII      000000    HH      HH   AA      AA   NN      NN  DDDDDDDD
PP           AA      AA  SSSSSSSS    IIIIII      000000    HH      HH   AA      AA   NN      NN  DDDDDDDD

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 PAS$$IO_HANDLER ( %TITLE 'I/O Error Handler'
;     2          0002  0                     IDENT = '1-003'         ! File: PASIOHAND.B32 Edit: SBL1003
;     3          0003  0                     ) =
;     4          0004  1 BEGIN
;     5          0005  1 !
;     6          0006  1 !*****************************************************************************
;     7          0007  1 !*                                                                           *
;     8          0008  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;     9          0009  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    10          0010  1 !*  ALL RIGHTS RESERVED.                                                      *
;    11          0011  1 !*                                                                           *
;    12          0012  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
;    13          0013  1 !*  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE           *
;    14          0014  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER      *
;    15          0015  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
;    16          0016  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY       *
;    17          0017  1 !*  TRANSFERRED.                                                              *
;    18          0018  1 !*                                                                           *
;    19          0019  1 !*  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE     *
;    20          0020  1 !*  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT         *
;    21          0021  1 !*  CORPORATION.                                                              *
;    22          0022  1 !*                                                                           *
;    23          0023  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS      *
;    24          0024  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;    25          0025  1 !*                                                                           *
;    26          0026  1 !*                                                                           *
;    27          0027  1 !*****************************************************************************
;    28          0028  1 !
;    29          0029  1
;    30          0030  1 !++
;    31          0031  1 ! FACILITY:     Pascal Language Support
;    32          0032  1 !
;    33          0033  1 ! ABSTRACT:
;    34          0034  1 !
;    35          0035  1 !       This module contains PAS$$IO_HANDLER, the condition handler established
;    36          0036  1 !       by all Pascal I/O procedures.
;    37          0037  1 !
;    38          0038  1 ! ENVIRONMENT:  User mode - AST reentrant
;    39          0039  1 !
;    40          0040  1 ! AUTHOR: Steven B. Lionel, CREATION DATE: 1-April-1981
;    41          0041  1 !
;    42          0042  1 ! MODIFIED BY:
;    43          0043  1 !
;    44          0044  1 ! 1-001 - Original.  SBL 1-April-1981
;    45          0045  1 ! 1-002 - If a zero-level access violation occurs while in PAS$$VALIDATE_PFV,
;    46          0046  1 !         convert the message to PAS$_INVFILVAR.  SBL 14-July-1982
;    47          0047  1 ! 1-003 - Check for PAS$_CONTINUE regardless of whether ERROR:=CONTINUE
;    48          0048  1 !         was specified.  SBL 29-Oct-1982
;    49          0049  1 !--
;    50          0050  1
```

```
;    52          0051  1 %SBTTL 'Declarations'
;    53          0052  1 !
;    54          0053  1 ! PROLOGUE DEFINITIONS:
;    55          0054  1 !
;    56          0055  1
;    57          0056  1 REQUIRE 'RTLIN:PASPROLOG';              ! Externals, linkages, PSECTs, structures
;    58          0120  1
;    59          0121  1 !
;    60          0122  1 ! TABLE OF CONTENTS:
;    61          0123  1 !
;    62          0124  1
;    63          0125  1 FORWARD ROUTINE
;    64          0126  1     PAS$$IO_HANDLER;                  ! I/O error handler
;    65          0127  1
;    66          0128  1 !
;    67          0129  1 ! MACROS:
;    68          0130  1 !
;    69          0131  1 !       NONE
;    70          0132  1 !
;    71          0133  1 ! EQUATED SYMBOLS:
;    72          0134  1 !
;    73          0135  1 !       NONE
;    74          0136  1 !
;    75          0137  1 ! FIELDS:
;    76          0138  1 !
;    77          0139  1 !       NONE
;    78          0140  1 !
;    79          0141  1 ! OWN STORAGE:
;    80          0142  1 !
;    81          0143  1 !       NONE
```

```
;    83          0144  1 %SBTTL 'PAS$$IO_HANDLER - I/O Error handler'
;    84          0145  1 GLOBAL ROUTINE PAS$$IO_HANDLER (                    ! I/O error handler
;    85          0146  1         SIGNAL_ARGS: REF BLOCK [, BYTE],           ! Signal arguments array
;    86          0147  1         MECH_ARGS: REF BLOCK [, BYTE],             ! Mechanism arguments array
;    87          0148  1         ENABLE_ARGS: REF VECTOR [, LONG]           ! Enable arguments array
;    88          0149  1     ) =
;    89          0150  1
;    90          0151  1 !++
;    91          0152  1 ! FUNCTIONAL DESCRIPTION:
;    92          0153  1 !
;    93          0154  1 !       This is the condition handler enabled by all VAX-11 Pascal I/O
;    94          0155  1 !       procedures.  It implements the ERROR:=CONTINUE specifier by
;    95          0156  1 !       unwinding to the address specified in the enable argument ERROR_ADDR.
;    96          0157  1 !
;    97          0158  1 !       Upon an unwind, if the file is not fully opened, it is closed.  The
;    98          0159  1 !       file is unlocked on unwind.
;    99          0160  1 !
;   100          0161  1 ! CALLING SEQUENCE:
;   101          0162  1 !
;   102          0163  1 !       status.wlc.v = PAS$$IO_HANDLER (SIGNAL_ARGS.rl.ra, MECH_ARGS.rl.ra
;   103          0164  1 !                                       , ENABLE_ARGS.rl.ra)
;   104          0165  1 !
;   105          0166  1 ! FORMAL PARAMETERS:
;   106          0167  1 !
;   107          0168  1 !       SIGNAL_ARGS     - The signal argument list.
;   108          0169  1 !
;   109          0170  1 !       MECH_ARGS       - The mechanism argument list.
;   110          0171  1 !
;   111          0172  1 !       ENABLE_ARGS     - An array with the following
;   112          0173  1 !                         format:
;   113          0174  1 !
;   114          0175  1 !                               +----------------+
;   115          0176  1 !       +----------------+      |   ENB_COUNT    |  <-- ENABLE_ARGS
;   116          0177  1 !       |   ENB_COUNT    |      +----------------+
;   117          0178  1 !       +----------------+      |  ENB_PFV_ADDR  |
;   118          0179  1 !       |  ENB_PFV_ADDR  |      +----------------+
;   119          0180  1 !       +----------------+      | ENB_UNWIND_ACT |
;   120          0181  1 !       | ENB_UNWIND_ACT |      +----------------+
;   121          0182  1 !       +----------------+      | ENB_ERROR_ADDR |
;   122          0183  1 !       | ENB_ERROR_ADDR |      +----------------+
;   123          0184  1 !       +----------------+
;   124          0185  1 !
;   125          0186  1 !                         ENB_COUNT is the count of following enable arguments.
;   126          0187  1 !                         The count is always at least 2.
;   127          0188  1 !
;   128          0189  1 !                         ENB_PFV_ADDR - If non-zero, the address of a longword
;   129          0190  1 !                         containing the PFV our establisher is operating on.
;   130          0191  1 !
;   131          0192  1 !                         ENB_UNWIND_ACT - Specifies the action
;   132          0193  1 !                         to take on an unwind.  The values are:
;   133          0194  1 !                          PAS$K_UNWIND_NOP - Do nothing
;   134          0195  1 !                          PAS$K_UNWIND_UNLOCK - Unlock PFV
;   135          0196  1 !
;   136          0197  1 !                         ENB_ERROR_ADDR - If present and non-zero, the address
;   137          0198  1 !                         of a longword containing the PC of an instruction to
;   138          0199  1 !                         unwind to.  This is present if the user specified
;   139          0200  1 !                         Error:=Continue in the I/O statement.
```

```
;  140          0201  1 ! IMPLICIT INPUTS:
;  141          0202  1 !
;  142          0203  1 !       The signaller's PFV placed as the first FAO argument in the primary
;  143          0204  1 !       signalled message.
;  144          0205  1 !
;  145          0206  1 ! IMPLICIT OUTPUTS:
;  146          0207  1 !
;  147          0208  1 !       May clear PFV$V_LOCK upon unwind.
;  148          0209  1 !
;  149          0210  1 ! ROUTINE VALUE:
;  150          0211  1 !
;  151          0212  1 !       SS$_RESIGNAL
;  152          0213  1 !
;  153          0214  1 ! SIDE EFFECTS:
;  154          0215  1 !
;  155          0216  1 !       May cause an unwind.
;  156          0217  1 !
;  157          0218  1 !       If a zero-level access violation occurs while in PAS$$VALIDATE_PFV,
;  158          0219  1 !       the signal is converted to PAS$_INVFILVAR.
;  159          0220  1 !
;  160          0221  1 !--
;  161          0222  1
;  162          0223  2     BEGIN
;  163          0224  2
;  164          0225  2     LITERAL
;  165          0226  2         ENB_COUNT = 0,          ! Count of enable arguments
;  166          0227  2         ENB_PFV_ADDR = 1,       ! Address of address of PFV
;  167          0228  2         ENB_UNWIND_ACT= 2,      ! Address of unwind action
;  168          0229  2         ENB_ERROR_ADDR = 3;     ! Address of address of unwind PC
;  169          0230  2
;  170          0231  2     BUILTIN
;  171          0232  2         ACTUALCOUNT;
;  172          0233  2
;  173          0234  2     !+
;  174          0235  2     ! Determine if this is an unwind.
;  175          0236  2     !-
;  176          0237  2
;  177          0238  2     IF .SIGNAL_ARGS [CHF$L_SIG_NAME] EQLU SS$_UNWIND
;  178          0239  2     THEN
;  179          0240  3         BEGIN
;  180          0241  3         IF ..ENABLE_ARGS [ENB_UNWIND_ACT] NEQ PAS$K_UNWIND_NOP
;  181          0242  3         THEN
;  182          0243  4             BEGIN
;  183          0244  4
;  184          0245  4             !+
;  185          0246  4             ! Get our establisher's PFV.
;  186          0247  4             !-
;  187          0248  4
;  188          0249  4             LOCAL
;  189          0250  4                 PFV: REF $PAS$PFV_FILE_VARIABLE;
;  190          0251  4             PFV = ..ENABLE_ARGS [ENB_PFV_ADDR]; ! Get PFV address
;  191          0252  4
;  192          0253  4             PFV [PFV$V_LOCK] = 0;       ! Unlock PFV
;  193          0254  3             END;
;  194          0255  3         END
;  195          0256  3
;  196          0257  2     ELSE
```

```
;  197          0258  2
;  198          0259  3          BEGIN
;  199          0260  3
;  200          0261  3          LOCAL
;  201          0262  3              COND_NAME: BLOCK [4, BYTE]; ! Primary condition name
;  202          0263  3
;  203          0264  3          !+
;  204          0265  3          ! Get primary condition name.
;  205          0266  3          !-
;  206          0267  3
;  207          0268  3          COND_NAME = .SIGNAL_ARGS [CHF$L_SIG_NAME];
;  208          0269  3
;  209          0270  3          !+
;  210          0271  3          ! If this is a zero-level access violation, and if the PC is inside
;  211          0272  3          ! PAS$$VALIDATE_PFV, we assume that we have been passed an invalid
;  212          0273  3          ! file variable.  Convert the signal to PAS$_INVFILVAR.
;  213          0274  3          !-
;  214          0275  3
;  215          0276  4          IF (.COND_NAME EQLU SS$_ACCVIO) AND (.MECH_ARGS [CHF$L_MCH_DEPTH] EQL 0)
;  216          0277  3          THEN
;  217          0278  3              !+
;  218          0279  3              ! Compare PC against range for PAS$$VALIDATE_PFV.
;  219          0280  3              !-
;  220          0281  3              IF (.SIGNAL_ARGS [16,0,32,0] GEQA PAS$$VALIDATE_PFV) AND
;  221          0282  4                 (.SIGNAL_ARGS [16,0,32,0] LSSA PAS$$VALIDATE_PFV_END)
;  222          0283  3              THEN
;  223          0284  4                  BEGIN
;  224          0285  4                  SIGNAL_ARGS [CHF$L_SIG_NAME] = PAS$_INVFILVAR;
;  225          0286  4                  SIGNAL_ARGS [8,0,32,0] = 1;      ! FAO count
;  226          0287  4                  SIGNAL_ARGS [12,0,32,0] = ..ENABLE_ARGS [ENB_PFV_ADDR];
;  227          0288  4                  COND_NAME = .SIGNAL_ARGS [CHF$L_SIG_NAME];
;  228          0289  3                  END;
;  229          0290  3
;  230          0291  3          !+
;  231          0292  3          ! Is this a PAS$ error?  If not, resignal.
;  232          0293  3          !-
;  233          0294  3
;  234          0295  3          IF .COND_NAME [STS$V_FAC_NO] NEQU PAS$_FACILITY
;  235          0296  3          THEN
;  236          0297  3              RETURN SS$_RESIGNAL;
;  237          0298  3
;  238          0299  3          !+
;  239          0300  3          ! See if the PFVs match.  The signaller's PFV is the
;  240          0301  3          ! first FAO parameter in the primary message.  If so, then
;  241          0302  3          ! continue with Pascal-specific processing.
;  242          0303  3          !-
;  243          0304  3
;  244          0305  3          IF .SIGNAL_ARGS [12,0,32,0] EQLA ..ENABLE_ARGS [ENB_PFV_ADDR]
;  245          0306  3          THEN
;  246          0307  4              BEGIN
;  247          0308  4
;  248          0309  4              !+
;  249          0310  4              ! Check to see if the current exception code is PAS$_CONTINUE.
;  250          0311  4              ! This is generated by PAS$$SIGNAL if a user condition handler
;  251          0312  4              ! continued from the signal.  If so, do a default unwind to
;  252          0313  4              ! the caller of our establisher.
;  253          0314  4              !-
```

```
;  254          0315  4
;  255          0316  4              IF .COND_NAME EQLU PAS$_CONTINUE
;  256          0317  4              THEN
;  257          0318  5                  IF NOT $UNWIND ()
;  258          0319  4                  THEN
;  259          0320  4                      $PAS$BUGCHECK (BUG_UNWINDFAIL);
;  260          0321  4
;  261          0322  4              !+
;  262          0323  4              ! Has ERROR:=CONTINUE been specified by our establisher?  This
;  263          0324  4              ! is determined by having ENB_ERROR_ADDR be non-zero.  If so,
;  264          0325  4              ! we may want to unwind.
;  265          0326  4              !-
;  266          0327  4
;  267          0328  4              IF .ENABLE_ARGS [ENB_COUNT] GEQU ENB_ERROR_ADDR
;  268          0329  4              THEN
;  269          0330  5                  BEGIN
;  270          0331  5                  IF ..ENABLE_ARGS [ENB_ERROR_ADDR] NEQ 0
;  271          0332  5                  THEN
;  272          0333  6                      BEGIN
;  273          0334  6
;  274          0335  6                      !+
;  275          0336  6                      ! See if the error message is one which is "trapped"
;  276          0337  6                      ! by ERROR:=CONTINUE.  This is done by comparing the
;  277          0338  6                      ! message number against a select range.
;  278          0339  6                      !-
;  279          0340  6
;  280          0341  6                      IF .COND_NAME [STS$V_CODE] GEQU PAS$$K_MSGCONTLO AND ! Lowest number
;  281          0342  6                         .COND_NAME [STS$V_CODE] LEQU PAS$$K_MSGCONTHI
;  282          0343  6                      THEN
;  283          0344  7                          BEGIN
;  284          0345  7
;  285          0346  7                          !+
;  286          0347  7                          ! We want to do the ERROR:=CONTINUE.  Unwind to the caller
;  287          0348  7                          ! of our establisher at the specified PC.
;  288          0349  7                          !-
;  289          0350  7
;  290          0351  8                          IF NOT $UNWIND (NEWPC = ..ENABLE_ARGS [ENB_ERROR_ADDR])
;  291          0352  7                          THEN
;  292          0353  7                              $PAS$BUGCHECK (BUG_UNWINDFAIL);
;  293          0354  6                          END;
;  294          0355  5                      END;
;  295          0356  4                  END;
;  296          0357  3              END;
;  297          0358  2          END;
;  298          0359  2
;  299          0360  2      RETURN SS$_RESIGNAL;                        ! Resignal error
;  300          0361  2
;  301          0362  1      END;                                        ! End of routine PAS$$IO_HANDLER


                                .TITLE  PAS$$IO_HANDLER I/O Error Handler
                                .IDENT  \1-003\

                                .EXTRN  PAS$$IO_HANDLER
                                .EXTRN  PAS$$VALIDATE_PFV
                                .EXTRN  PAS$$VALIDATE_PFV_END
                                .EXTRN  PAS$_INVFILVAR, PAS$_FACILITY
```

```
                                                                      .EXTRN  PAS$_CONTINUE, SYS$UNWIND
                                                                      .EXTRN  PAS$$BUGCHECK, PAS$$K_MSGCONTLO
                                                                      .EXTRN  PAS$$K_MSGCONTHI

                                                                      .PSECT  _PAS$CODE,NOWRT,  SHR,  PIC,2

                                                     001C 00000       .ENTRY  PAS$$IO_HANDLER, Save R2,R3,R4                 ; 0145
                                54 00000000G   00   9E 00002          MOVAB   SYS$UNWIND, R4                                 ;
                                52        0C   AC   D0 00009          MOVL    ENABLE_ARGS, R2                                ; 0241
                                50        04   AC   D0 0000D          MOVL    SIGNAL_ARGS, R0                                ; 0238
                      00000920  8F        04   A0   D1 00011          CMPL    4(R0), #2336                                   ;
                                               11   12 00019          BNEQ    2$                                             ;
                                          08   B2   D5 0001B          TSTL    @8(R2)                                         ; 0241
                                               09   13 0001E          BEQL    1$                                             ;
                                51        04   B2   D0 00020          MOVL    @4(R2), PFV                                    ; 0251
                            07  A1        80   8F   8A 00024          BICB2   #128, 7(PFV)                                   ; 0253
                                             009A   31 00029  1$:     BRW     6$                                             ; 0238
                                53        04   A0   D0 0002C  2$:     MOVL    4(R0), COND_NAME                               ; 0268
                                0C             53   D1 00030          CMPL    COND_NAME, #12                                 ; 0276
                                               3C   12 00033          BNEQ    3$                                             ;
                                51        08   AC   D0 00035          MOVL    MECH_ARGS, R1                                  ;
                                          08   A1   D5 00039          TSTL    8(R1)                                          ;
                                               33   12 0003C          BNEQ    3$                                             ;
                                51 00000000G   00   9E 0003E          MOVAB   PAS$$VALIDATE_PFV, R1                          ; 0281
                                51        10   A0   D1 00045          CMPL    16(R0), R1                                     ;
                                               26   1F 00049          BLSSU   3$                                             ;
                                51 00000000G   00   9E 0004B          MOVAB   PAS$$VALIDATE_PFV_END, R1                      ; 0282
                                51        10   A0   D1 00052          CMPL    16(R0), R1                                     ;
                                               19   1E 00056          BGEQU   3$                                             ;
                            04  A0 00000000G   8F   D0 00058          MOVL    #PAS$_INVFILVAR, 4(R0)                         ; 0285
                            08  A0             01   D0 00060          MOVL    #1, 8(R0)                                      ; 0286
                                51        0C   AC   D0 00064          MOVL    ENABLE_ARGS, R1                                ; 0287
                            0C  A0        04   B1   D0 00068          MOVL    @4(R1), 12(R0)                                 ;
                                53        04   A0   D0 0006D          MOVL    4(R0), COND_NAME                               ; 0288
         00G            53      0C             10   ED 00071  3$:     CMPZV   #16, #12, COND_NAME, S^PAS$_FACILITY           ; 0295
                                               4E   12 00076          BNEQ    6$                                             ;
                            04  B2        0C   A0   D1 00078          CMPL    12(R0), @4(R2)                                 ; 0305
                                               47   12 0007D          BNEQ    6$                                             ;
                      00000000G 8F             53   D1 0007F          CMPL    COND_NAME, #PAS$_CONTINUE                      ; 0316
                                               08   12 00086          BNEQ    4$                                             ;
                                               7E   7C 00088          CLRQ    -(SP)                                          ; 0318
                                64             02   FB 0008A          CALLS   #2, SYS$UNWIND                                 ;
                                2B             50   E9 0008D          BLBC    R0, 5$                                         ;
                                03             62   D1 00090  4$:     CMPL    (R2), #3                                       ; 0328
                                               31   1F 00093          BLSSU   6$                                             ;
                                          0C   B2   D5 00095          TSTL    @12(R2)                                        ; 0331
                                               2C   13 00098          BEQL    6$                                             ;
00000000G  8F           53      0C             03   ED 0009A          CMPZV   #3, #12, COND_NAME, #PAS$$K_MSGCONTLO          ; 0341
                                               21   1F 000A3          BLSSU   6$                                             ;
00000000G  8F           53      0C             03   ED 000A5          CMPZV   #3, #12, COND_NAME, #PAS$$K_MSGCONTHI          ; 0342
                                               16   1A 000AE          BGTRU   6$                                             ;
                                          0C   B2   DD 000B0          PUSHL   @12(R2)                                        ; 0351
                                               7E   D4 000B3          CLRL    -(SP)                                          ;
                                64             02   FB 000B5          CALLS   #2, SYS$UNWIND                                 ;
                                0B             50   E8 000B8          BLBS    R0, 6$                                         ;
                                               03   DD 000BB  5$:     PUSHL   #3                                             ; 0353
                      00000000G 00             01   FB 000BD          CALLS   #1, PAS$$BUGCHECK                              ;
```

```
                                06  11 000C4         BRB     7$                              ;
              50      0918      8F  3C 000C6 6$:     MOVZWL  #2328, R0                       ; 0360
                                    04 000CB         RET                                     ;
                                50  D4 000CC 7$:     CLRL    R0                              ; 0362
                                    04 000CE         RET                                     ;

; Routine Size:  207 bytes,    Routine Base:  _PAS$CODE + 0000


;  302          0363  1
```

```
;  304          0364  1 END                                   ! End of module PAS$$IO_HANDLER
;  305          0365  1
;  306          0366  0 ELUDOM
```

```
;                                   PSECT SUMMARY

;       Name                    Bytes                           Attributes

;  _PAS$CODE                        207  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
```

```
;                                 Library Statistics

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

;  _$255$DUA28:[SYSLIB]STARLET.L32;1         9776          9          0         581       00:01.0
;  _$255$DUA28:[PASRTL.OBJ]PASLIB.L32;1       427         32          7          33       00:00.4
```

```
;                                    COMMAND QUALIFIERS

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

; Size:          207 code + 0 data bytes
; Run Time:        00:07.0
; Elapsed Time:    00:31.1
; Lines/CPU Min:    3150
; Lexemes/CPU-Min: 12086
; Memory Used:  91 pages
; Compilation Complete
```

PASPAGE2
LIS

PASODD
LIS

PASHEX
LIS

PASLINELI
LIS

PASMSGPTR
LIS

PASLIB
LIS

PASLOOKAH
LIS

PASOCT
LIS

PASOPEN2
LIS

PASPUT
LIS

PASTOHAND
LIS

PASLOCATE
LIS

PASMSGTXT
LIS