```
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  HH      HH    AAAAAA    NN      NN  DDDDDDDD    LL          EEEEEEEEEE
PPPPPPPP      AAAAAA      SSSSSSSS  HH      HH    AAAAAA    NN      NN  DDDDDDDD    LL          EEEEEEEEEE
PP      PP   AA      AA  SS        HH      HH   AA      AA   NN      NN  DD      DD  LL          EE
PP      PP   AA      AA  SS        HH      HH   AA      AA   NN      NN  DD      DD  LL          EE
PP      PP   AA      AA  SS        HH      HH   AA      AA   NNNN    NN  DD      DD  LL          EE
PP      PP   AA      AA  SS        HH      HH   AA      AA   NNNN    NN  DD      DD  LL          EE
PPPPPPPP     AA      AA    SSSSSS  HHHHHHHHHH   AA      AA   NN  NN  NN  DD      DD  LL          EEEEEEEE
PPPPPPPP     AA      AA    SSSSSS  HHHHHHHHHH   AA      AA   NN  NN  NN  DD      DD  LL          EEEEEEEE
PP           AAAAAAAAAA          SS  HH      HH  AAAAAAAAAA   NN    NNNN  DD      DD  LL          EE
PP           AAAAAAAAAA          SS  HH      HH  AAAAAAAAAA   NN    NNNN  DD      DD  LL          EE
PP           AA      AA          SS  HH      HH  AA      AA   NN      NN  DD      DD  LL          EE                ....
PP           AA      AA          SS  HH      HH  AA      AA   NN      NN  DD      DD  LL          EE                ....
PP           AA      AA  SSSSSSSS  HH      HH  AA      AA   NN      NN  DDDDDDDD    LLLLLLLLLL  EEEEEEEEEE        ....
PP           AA      AA  SSSSSSSS  HH      HH  AA      AA   NN      NN  DDDDDDDD    LLLLLLLLLL  EEEEEEEEEE        ....


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$HANDLER ( %TITLE 'Handler established by compiled code'
;     2          0002  0                     IDENT = '1-002'         ! File: PASANDLE.B32 Edit: SBL1002
;     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$HANDLER, the condition handler established
;    36          0036  1 !       by Pascal compiled code routines which have local files or checking.
;    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 - Add enable of PAS$$GOTO_HANDLER, call of PAS$$UNWIND_GOTO to
;    46          0046  1 !         support revised PAS$GOTO.  SBL 28-Jan-1983
;    47          0047  1 !--
;    48          0048  1
```

```
;    50          0049  1 %SBTTL 'Declarations'
;    51          0050  1 !
;    52          0051  1 ! PROLOGUE DEFINITIONS:
;    53          0052  1 !
;    54          0053  1
;    55          0054  1 REQUIRE 'RTLIN:PASPROLOG';                  ! Externals, linkages, PSECTs, structures
;    56          0118  1
;    57          0119  1 !
;    58          0120  1 ! TABLE OF CONTENTS:
;    59          0121  1 !
;    60          0122  1
;    61          0123  1 FORWARD ROUTINE
;    62          0124  1     PAS$HANDLER;                            ! Compiled code handler
;    63          0125  1
;    64          0126  1 !
;    65          0127  1 ! MACROS:
;    66          0128  1 !
;    67          0129  1 !       NONE
;    68          0130  1 !
;    69          0131  1 ! EQUATED SYMBOLS:
;    70          0132  1 !
;    71          0133  1 !       NONE
;    72          0134  1 !
;    73          0135  1 ! FIELDS:
;    74          0136  1 !
;    75          0137  1 !       NONE
;    76          0138  1 !
;    77          0139  1 ! OWN STORAGE:
;    78          0140  1 !
;    79          0141  1 !       NONE
```

```
;    81          0142  1 %SBTTL 'PAS$HANDLER - Compiled code handler'
;    82          0143  1 GLOBAL ROUTINE PAS$HANDLER (
;    83          0144  1         SIGNAL_ARGS: REF BLOCK [, BYTE],        . Signal arguments array
;    84          0145  1         MECH_ARGS: REF BLOCK [, BYTE]           . Mechanism arguments array
;    85          0146  1     ) =
;    86          0147  1
;    87          0148  1 !++
;    88          0149  1 ! FUNCTIONAL DESCRIPTION:
;    89          0150  1 !
;    90          0151  1 !       This condition handler is established by VAX-11 Pascal compiled
;    91          0152  1 !       code in the procedure prologue.  It performs the following
;    92          0153  1 !       services:
;    93          0154  1 !
;    94          0155  1 !          If the current exception is PAS$_GOTO, generated by an up-level
;    95          0156  1 !          GOTO, PAS$$UNWIND_GOTO is called and control returns to CHF.
;    96          0157  1 !
;    97          0158  1 !          Otherwise:
;    98          0159  1 !               1.  Calls user condition handler declared with
;    99          0160  1 !                   ESTABLISH, including the environment value.
;   100          0161  1 !                   The user handler address and the environment
;   101          0162  1 !                   value are at offsets -8 and -4 from the establisher's FP.
;   102          0163  1 !
;   103          0164  1 !               2.  Intercepts certain exceptions generated by compiled
;   104          0165  1 !                   code for run-time error checking, and converts them
;   105          0166  1 !                   into Pascal-specific messages.
;   106          0167  1 !
;   107          0168  1 !               3.  Closes any locally-declared files upon an unwind.
;   108          0169  1 !
;   109          0170  1 ! CALLING SEQUENCE:
;   110          0171  1 !
;   111          0172  1 !       status.wlc.v = PAS$HANDLER (SIGNAL_ARGS.ml.ra, MECH_ARGS.rl.ra)
;   112          0173  1 !
;   113          0174  1 ! FORMAL PARAMETERS:
;   114          0175  1 !
;   115          0176  1 !       SIGNAL_ARGS     - The signal argument list.  This may be modified.
;   116          0177  1 !
;   117          0178  1 !       MECH_ARGS       - The mechanism argument list.
;   118          0179  1 !
;   119          0180  1 ! IMPLICIT INPUTS:
;   120          0181  1 !
;   121          0182  1 !       For certain exceptions, the instruction stream pointed to by the
;   122          0183  1 !       signaled PC is used to determine the correct Pascal-specific
;   123          0184  1 !       message.
;   124          0185  1 !
;   125          0186  1 !       User-established condition handler and environment value at -8 and
;   126          0187  1 !       -4 of users FP.
;   127          0188  1 !
;   128          0189  1 ! IMPLICIT OUTPUTS:
;   129          0190  1 !
;   130          0191  1 !       NONE
;   131          0192  1 !
;   132          0193  1 ! ROUTINE VALUE:
;   133          0194  1 !
;   134          0195  1 !       SS$_RESIGNAL, or as returned from user handler
;   135          0196  1 !
;   136          0197  1 ! SIDE EFFECTS:
;   137          0198  1 !
```

```
;  138          0199  1 !       May close local files.
;  139          0200  1 !
;  140          0201  1 !--
;  141          0202  1
;  142          0203  2     BEGIN
;  143          0204  2
;  144          0205  2     LOCAL
;  145          0206  2         USERS_FRAME: REF VECTOR [, LONG],       ! User's stack frame
;  146          0207  2         RETURN_STATUS;                          ! Handler return status
;  147          0208  2
;  148          0209  2     LABEL
;  149          0210  2         INSTR_BLOCK;                            ! Block looking at instruction stream
;  150          0211  2
;  151          0212  2     BUILTIN
;  152          0213  2         AP,                                     ! Argument pointer
;  153          0214  2         CALLG,                                  ! CALLG instruction
;  154          0215  2         PROBER;                                 ! Probe for read
;  155          0216  2
;  156          0217  2     !+
;  157          0218  2     ! Declare special linkage used for calling user condition handlers.
;  158          0219  2     !-
;  159          0220  2
;  160          0221  2     LINKAGE
;  161          0222  2         CALL_USER_HANDLER = CALL (REGISTER=1,   ! Environment pointer
;  162          0223  2                                   STANDARD,     ! Signal argument list
;  163          0224  2                                   STANDARD);    ! Mechanism argument list
;  164          0225  2     !+
;  165          0226  2     ! Establish PAS$$GOTO_HANDLER as our handler to catch up-level GOTOs
;  166          0227  2     ! while doing condition handling.
;  167          0228  2     !-
;  168          0229  2
;  169          0230  2     ENABLE
;  170          0231  2         PAS$$GOTO_HANDLER;
;  171          0232  2
;  172          0233  2     !+
;  173          0234  2     ! Get the stack frame pointer of our establisher.
;  174          0235  2     !-
;  175          0236  2
;  176          0237  2     USERS_FRAME = .MECH_ARGS [CHF$L_MCH_FRAME];
;  177          0238  2
;  178          0239  2     !+
;  179          0240  2     ! Initialize handler return status to SS$_RESIGNAL.
;  180          0241  2     !-
;  181          0242  2
;  182          0243  2     RETURN_STATUS = SS$_RESIGNAL;
;  183          0244  2
;  184          0245  2     !+
;  185          0246  2     ! Determine if this is an up-level GOTO.  If so, call PAS$$UNWIND_GOTO
;  186          0247  2     ! and return.  It may only return SS$_RESIGNAL, but we don't want
;  187          0248  2     ! the user handlers to see PAS$_GOTO.
;  188          0249  2     !-
;  189          0250  2
;  190          0251  2     IF .SIGNAL_ARGS [CHF$L_SIG_NAME] EQLU PAS$_GOTO
;  191          0252  2     THEN
;  192          0253  2         RETURN (CALLG (.AP, PAS$$UNWIND_GOTO));
;  193          0254  2
;  194          0255  2     !+
```

```
;  195          0256  2      ! Determine if this is an unwind.
;  196          0257  2      !-
;  197          0258  2
;  198          0259  2      IF .SIGNAL_ARGS [CHF$L_SIG_NAME] EQLU SS$_UNWIND
;  199          0260  2      THEN
;  200          0261  3          BEGIN
;  201          0262  3
;  202          0263  3          !+
;  203          0264  3          ! This is an unwind.  First, call the user's handler, if there
;  204          0265  3          ! is one.
;  205          0266  3          !-
;  206          0267  3
;  207          0268  3          IF .USERS_FRAME [-2] NEQ 0      ! Handler present?
;  208          0269  3          THEN
;  209          0270  3              RETURN_STATUS = CALL_USER_HANDLER (
;  210          0271  3                  .USERS_FRAME [-2],      ! Handler address
;  211          0272  3                  .USERS_FRAME [-1],      ! Environment
;  212          0273  3                  .SIGNAL_ARGS,           ! Signal arguments
;  213          0274  3                  .MECH_ARGS);            ! Mechanism arguments
;  214          0275  3
;  215          0276  3          !+
;  216          0277  3          ! Now call PAS$$CLOSE_LOCAL to close any local files.
;  217          0278  3          !-
;  218          0279  3
;  219          0280  3          PAS$$CLOSE_LOCAL (.USERS_FRAME);
;  220          0281  3
;  221          0282  3          END     ! End of actions for unwind
;  222          0283  3
;  223          0284  2      ELSE
;  224          0285  2
;  225          0286  3          BEGIN
;  226          0287  3
;  227          0288  3          !+
;  228          0289  3          ! This is not an unwind.  Check to see if it is an exception generated
;  229          0290  3          ! by compiled code.  If so, information in the instruction stream
;  230          0291  3          ! tells us which PAS$ message to turn the signal into.
;  231          0292  3          !
;  232          0293  3          ! The following table gives the combinations we are interested in,
;  233          0294  3          ! and the location of the code (0-63)
;  234          0295  3          !       Exception       Instruction     Code byte location
;  235          0296  3          !       ---------       -----------     ------------------
;  236          0297  3          !       SS$_OPCDEC      HALT            .BYTE after HALT
;  237          0298  3          !       SS$_SUBRNG      INDEX           TSTB #x (current instruction)
;  238          0299  3          !       SS$_INTOVF      any             TSTB #x (current instruction)
;  239          0300  3          !
;  240          0301  3          ! Special case:  If the exception is SS$_SUBRNG and the next instruction
;  241          0302  3          ! is not a TSTB #x, signal PAS$_ARRINDVAL.
;  242          0303  3          !-
;  243          0304  3
;  244          0305  3          IF  .MECH_ARGS [CHF$L_MCH_DEPTH] EQL 0 AND
;  245          0306  4              ((.SIGNAL_ARGS [CHF$L_SIG_NAME] EQLU SS$_OPCDEC) OR
;  246          0307  4               (.SIGNAL_ARGS [CHF$L_SIG_NAME] EQLU SS$_SUBRNG) OR
;  247          0308  4               (.SIGNAL_ARGS [CHF$L_SIG_NAME] EQLU SS$_INTOVF))
;  248          0309  3          THEN
;  249          0310  3              !+
;  250          0311  3              ! Create labelled block to contain code that looks at the
;  251          0312  3              ! instruction stream.  If at any time, the conditions are not
```

```
;  252          0313  3              ! consistent with converting the exception to a PAS$ exception,
;  253          0314  3              ! just leave the block.
;  254          0315  3              !-
;  255          0316  4              INSTR_BLOCK: BEGIN
;  256          0317  4
;  257          0318  4              LOCAL
;  258          0319  4                  INSTR_PC: REF VECTOR [, BYTE],  ! Signaled PC
;  259          0320  4                  OPCODE: WORD,                   ! Opcode at signaled PC
;  260          0321  4                  NEW_MESSAGE: BLOCK [4, BYTE];   ! Replacement message code
;  261          0322  4
;  262          0323  4              LITERAL
;  263          0324  4                  OPC_HALT = %X'00',              ! Opcode for HALT
;  264          0325  4                  OPC_BPT = %X'03',               ! Opcode for BPT
;  265          0326  4                  OPC_TSTB = %X'95';              ! Opcode for TSTB
;  266          0327  4
;  267          0328  4              !+
;  268          0329  4              ! Are there just 3 signal arguments?  There should be just the
;  269          0330  4              ! condition name, PC and PSL.  If not, resignal.
;  270          0331  4              !-
;  271          0332  4
;  272          0333  4              IF .SIGNAL_ARGS [CHF$L_SIG_ARGS] NEQ 3
;  273          0334  4              THEN
;  274          0335  4                  LEAVE INSTR_BLOCK;              ! Just resignal
;  275          0336  4
;  276          0337  4              !+
;  277          0338  4              ! Can we read the instruction byte and the following byte?
;  278          0339  4              !-
;  279          0340  4
;  280          0341  4              INSTR_PC = .SIGNAL_ARGS [8,0,32,0]; ! Get PC of instruction                ;
;  281          0342  4              IF NOT PROBER (%REF(0), %REF(2), INSTR_PC [0])      ! PC
;  282          0343  4              THEN
;  283          0344  4                  LEAVE INSTR_BLOCK;              ! Resignal if can't access             ;
;  284          0345  4
;  285          0346  4              !+
;  286          0347  4              ! Get the opcode byte.
;  287          0348  4              !-
;  288          0349  4
;  289          0350  4              OPCODE = .INSTR_PC [0];
;  290          0351  4
;  291          0352  4              !+
;  292          0353  4              ! If the opcode byte is BPT, then it looks like the user
;  293          0354  4              ! set a breakpoint on this instruction.  Call LIB$GET_OPCODE
;  294          0355  4              ! to find out what the byte really is.
;  295          0356  4              !-
;  296          0357  4
;  297          0358  4 !! Don't call LIB$GET_OPCODE on VMS V2
;  298        L 0359  4 %IF %VARIANT
;  299        U 0360  4 %THEN
;  300          0361  4 %ELSE
;  301          0362  4              IF .OPCODE EQLU OPC_BPT     ! BPT opcode
;  302          0363  4              THEN
;  303          0364  4                  OPCODE = LIB$GET_OPCODE (INSTR_PC [0]);
;  304          0365  4 %FI
;  305          0366  4
;  306          0367  4              !+
;  307          0368  4              ! Now, if the exception is SS$_OPCDEC, then the instruction
;  308          0369  4              ! must be HALT.  Otherwise, it must be TSTB.
```

```
;  309          0370  4               !-
;  310          0371  4
;  311          0372  4               IF .SIGNAL_ARGS [CHF$L_SIG_NAME] EQL SS$_OPCDEC
;  312          0373  4               THEN
;  313          0374  5                   BEGIN
;  314          0375  5                   IF .OPCODE NEQ OPC_HALT
;  315          0376  5                   THEN
;  316          0377  5                       LEAVE INSTR_BLOCK;          ! Just resignal
;  317          0378  5                   END
;  318          0379  4               ELSE IF .OPCODE NEQ OPC_TSTB        ! SUBRNG or INTOVF
;  319          0380  4               THEN
;  320          0381  5                   BEGIN
;  321          0382  5                   IF .SIGNAL_ARGS [CHF$L_SIG_NAME] EQL SS$_SUBRNG
;  322          0383  5                   THEN
;  323          0384  5                       SIGNAL_ARGS [CHF$L_SIG_NAME] = PAS$_ARRINDVAL;
;  324          0385  5                   LEAVE INSTR_BLOCK;              ! Just resignal
;  325          0386  4                   END;
;  326          0387  4
;  327          0388  4               !+
;  328          0389  4               ! Ok, we now know that the current instruction is
;  329          0390  4               ! one that the compiler uses to signal us that we should
;  330          0391  4               ! translate the exception.  The next byte must be in the
;  331          0392  4               ! range 0-63.  This is either a .BYTE following the
;  332          0393  4               ! HALT or a short literal following the TSTB.
;  333          0394  4               !-
;  334          0395  4
;  335          0396  4               IF .INSTR_PC [1] GTRU 63
;  336          0397  4               THEN
;  337          0398  4                   LEAVE INSTR_BLOCK;              ! Just resignal
;  338          0399  4
;  339          0400  4               !+
;  340          0401  4               ! If the code is PAS$K_IGNORE, meaning that the compiled
;  341          0402  4               ! code wants us to ignore this exception, continue execution
;  342          0403  4               ! by returning SS$_CONTINUE.
;  343          0404  4               !-
;  344          0405  4
;  345          0406  4               IF .INSTR_PC [1] EQLU PAS$K_IGNORE
;  346          0407  4               THEN
;  347          0408  4                   RETURN SS$_CONTINUE;
;  348          0409  4
;  349          0410  4               !+
;  350          0411  4               ! Ok, construct the PAS$ message, replacing the system
;  351          0412  4               ! message.
;  352          0413  4               !-
;  353          0414  4
;  354          0415  4               NEW_MESSAGE = PAS$_BASE;                ! Base message
;  355          0416  4               NEW_MESSAGE [STS$V_CODE] = .INSTR_PC [1] + PAS$$K_MSGCHKBAS;
;  356          0417  4               SIGNAL_ARGS [CHF$L_SIG_NAME] = .NEW_MESSAGE;
;  357          0418  4
;  358          0419  4               !+
;  359          0420  4               ! Step the PC over the two bytes.
;  360          0421  4               !-
;  361          0422  4
;  362          0423  4               SIGNAL_ARGS [8,0,32,0] = .SIGNAL_ARGS [8,0,32,0] + 2;
;  363          0424  4
;  364          0425  3               END;
;  365          0426  3
```

```
;  366          0427  3          !+
;  367          0428  3          ! Call the user's handler, if any.
;  368          0429  3          !-
;  369          0430  3
;  370          0431  3          IF .USERS_FRAME [-2] NEQ 0      ! Handler present?
;  371          0432  3          THEN
;  372          0433  3              RETURN_STATUS = CALL_USER_HANDLER (
;  373          0434  3                  .USERS_FRAME [-2],      ! Handler address
;  374          0435  3                  .USERS_FRAME [-1],      ! Environment
;  375          0436  3                  .SIGNAL_ARGS,           ! Signal arguments
;  376          0437  3                  .MECH_ARGS);            ! Mechanism arguments
;  377          0438  3
;  378          0439  2          END;
;  379          0440  2
;  380          0441  2      RETURN .RETURN_STATUS;                      ! Return to CHF
;  381          0442  2
;  382          0443  1      END;                                        ! End of routine PAS$HANDLER


                                                                .TITLE  PAS$HANDLER Handler established by compiled cod
                                                                        e
;                                                               .IDENT  \1-002\

                                                                .EXTRN  PAS$HANDLER, PAS$$GOTO_HANDLER
                                                                .EXTRN  PAS$_GOTO, PAS$$UNWIND_GOTO
                                                                .EXTRN  PAS$$CLOSE_LOCAL
                                                                .EXTRN  LIB$GET_OPCODE, PAS$_ARRINDVAL
                                                                .EXTRN  PAS$K_IGNORE, PAS$_BASE
                                                                .EXTRN  PAS$$R_MSGCHKBAS

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

                                                01FC 00000      .ENTRY  PAS$HANDLER, Save R2,R3,R4,R5,R6,R7,R8      ; 0143
                                            6D      010D    CF  DE 00002        MOVAL   12$, (FP)                   ; 0203
                                            50        08    AC  D0 00007        MOVL    MECH_ARGS, R0               ; 0237
                                            55        04    A0  D0 0000B        MOVL    4(R0), USERS_FRAME          ;
                                            58      0918    8F  3C 0000F        MOVZWL  #2328, RETURN_STATUS        ; 0243
                                            50        04    AC  D0 00014        MOVL    SIGNAL_ARGS, R0             ; 0251
                                00000000G   8F        04    A0  D1 00018        CMPL    4(R0), #PAS$_GOTO           ;
                                                        08  12 00020        BNEQ    1$                          ;
                                00000000G   00              6C  FA 00022        CALLG   (AP), PAS$$UNWIND_GOTO      ; 0253
                                                            04 00029        RET                                 ;
                                            54        04    AC  D0 0002A 1$:    MOVL    SIGNAL_ARGS, R4             ; 0259
                                            56        04    A4  D0 0002E        MOVL    4(R4), R6                   ;
                                00000920    8F              56  D1 00032        CMPL    R6, #2336                   ;
                                                            23  12 00039        BNEQ    3$                          ;
                                                    F8      A5  D5 0003B        TSTL    -8(USERS_FRAME)             ; 0268
                                                            10  13 0003E        BEQL    2$                          ;
                                                    08      AC  DD 00040        PUSHL   MECH_ARGS                   ; 0274
                                                            54  DD 00043        PUSHL   R4                          ; 0273
                                            51        FC    A5  D0 00045        MOVL    -4(USERS_FRAME), R1         ; 0272
                                        F8  B5              02  FB 00049        CALLS   #2, @-8(USERS_FRAME)        ;
                                            58              50  D0 0004D        MOVL    R0, RETURN_STATUS           ;
                                                            55  DD 00050 2$:    PUSHL   USERS_FRAME                 ; 0280
                                                00000000G   00  16 00052        JSB     PAS$$CLOSE_LOCAL            ;
                                            5E              04  C0 00058        ADDL2   #4, SP                      ;
                                                        0081    31 0005B        BRW     11$                         ; 0259
```

```
                                57        08    AC  D0 0005E 3$:     MOVL    MECH_ARGS, R7                           ; 030
                                          08    A7  D5 00062          TSTL    8(R7)                                   ;
                                                65  12 00065          BNEQ    7$                                      ;
                         0000043C 8F              56  D1 00067          CMPL    R6, #1084                               ; 0306
                                                  12  13 0006E          BEQL    4$                                      ;
                         000004AC 8F              56  D1 00070          CMPL    R6, #1196                               ; 0307
                                                  09  13 00077          BEQL    4$                                      ;
                         0000047C 8F              56  D1 00079          CMPL    R6, #1148                               ; 0308
                                                  79  12 00080          BNEQ    10$                                     ;
                                03                64  D1 00082 4$:      CMPL    (R4), #3                                ; 0333
                                                  74  12 00085          BNEQ    10$                                     ;
                                52        08      A4  D0 00087          MOVL    8(R4), INSTR_PC                         ; 0341
                      62        02              00  0C 0008B          PROBER  #0, #2, (INSTR_PC)                      ; 0342
                                                  6A  13 0008F          BEQL    10$                                     ;
                                53                62  9B 00091          MOVZBW  (INSTR_PC), OPCODE                      ; 0350
                                03                53  B1 00094          CMPW    OPCODE, #3                              ; 0362
                                                  0C  12 00097          BNEQ    5$                                      ;
                                                  52  DD 00099          PUSHL   INSTR_PC                                ; 0364
                         00000000G 00             01  FB 0009B          CALLS   #1, LIB$GET_OPCODE                      ;
                                53                50  B0 000A2          MOVW    R0, OPCODE                              ;
                         0000043C 8F              56  D1 000A5 5$:      CMPL    R6, #1084                               ; 0372
                                                  06  12 000AC          BNEQ    6$                                      ;
                                                  53  B5 000AE          TSTW    OPCODE                                  ; 0375
                                                  1C  13 000B0          BEQL    8$                                      ;
                                                  47  11 000B2          BRB     10$                                     ; 0377
                             0095 8F              53  B1 000B4 6$:      CMPW    OPCODE, #149                            ; 0379
                                                  13  13 000B9          BEQL    8$                                      ;
                         000004AC 8F              56  D1 000BB          CMPL    R6, #1196                               ; 0382
                                                  37  12 000C2          BNEQ    10$                                     ;
                               04 A4 00000000G    8F  D0 000C4          MOVL    #PAS$_ARRINDVAL, 4(R4)                  ; 0384
                                                  2D  11 000CC 7$:      BRB     10$                                     ; 0385
                                3F        01      A2  91 000CE 8$:      CMPB    1(INSTR_PC), #63                        ; 0396
                                                  27  1A 000D2          BGTRU   10$                                     ;
                                00G       01      A2  91 000D4          CMPB    1(INSTR_PC), S^PAS$K_IGNORE             ; 0406
                                                  04  12 000D8          BNEQ    9$                                      ;
                                50                01  D0 000DA          MOVL    #1, R0                                  ; 0408
                                                  04  000DD            RET                                             ;
                                50 00000000G      8F  D0 000DE 9$:      MOVL    #PAS$_BASE, NEW_MESSAGE                 ; 0415
                                51        01      A2  9A 000E5          MOVZBL  1(INSTR_PC), R1                         ; 0416
                                51      0000G     8F  A0 000E9          ADDW2   #PAS$$K_MSGCHKBAS, R1                   ;
                      50        0C      03        51  F0 000EE          INSV    R1, #3, #12, NEW_MESSAGE                ;
                               04 A4              50  D0 000F3          MOVL    NEW_MESSAGE, 4(R4)                      ; 0417
                               08 A4              02  C0 000F7          ADDL2   #2, 8(R4)                               ; 0423
                                          F8      A5  D5 000FB 10$:     TSTL    -8(USERS_FRAME)                         ; 0431
                                                  0F  13 000FE          BEQL    11$                                     ;
                                        0090      8F  BB 00100          PUSHR   #^M<R4,R7>                              ; 0436
                                51        FC      A5  D0 00104          MOVL    -4(USERS_FRAME), R1                     ; 0435
                               F8 B5              02  FB 00108          CALLS   #2, @-8(USERS_FRAME)                    ;
                                58                50  D0 0010C          MOVL    R0, RETURN_STATUS                       ;
                                50                58  D0 0010F 11$:     MOVL    RETURN_STATUS, R0                       ; 0441
                                                      04 00112          RET                                             ; 0443
                                                    0000 00113 12$:     .WORD   Save nothing                            ; 0203
                                                  7E  D4 00115          CLRL    -(SP)                                   ;
                                                  5E  DD 00117          PUSHL   SP                                      ;
                                7E        04      AC  7D 00119          MOVQ    4(AP), -(SP)                            ;
                         00000000G 00             03  FB 0011D          CALLS   #3, PAS$$GOTO_HANDLER                   ;
                                                      04 00124          RET                                             ;
```

```
; Routine Size:  293 bytes,    Routine Base:  _PAS$CODE + 0000

;  383          0444  1
;  384          0445  1 !<BLF/PAGE>
```

```
;  386          0446  1 END                                    . End of module PAS$HANDLER
;  387          0447  1
;  388          0448  0 ELUDOM
```

```
;                                  PSECT SUMMARY
;
;      Name                    Bytes                           Attributes
;
;  _PAS$CODE                       293  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        11         0         581        00:01.0
;  _$255$DUA28:[PASRTL.OBJ]PASLIB.L32;1      427        11         2          33        00:00.4
```

```
;                                     COMMAND QUALIFIERS

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

; Size:          293 code + 0 data bytes
; Run Time:          00:08.0
; Elapsed Time:      00:31.9
; Lines/CPU Min:      3364
; Lexemes/CPU-Min:  12000
; Memory Used:   105 pages
; Compilation Complete
```

PASFVOUTP
LIS

PASEOLN2
LIS

PASHEAP
LIS

PASHANDLE
LIS

PASFAB
LIS

PASGET
LIS

PASCVTRT
LIS

PASDATE
LIS

PASEOF2
LIS

PASFINDK
LIS

PASFVINPU
LIS

PASEXPO
LIS

PASGOTO
LIS

PASFILEUT
LIS

PASHALT
LIS

PASDELETE
LIS

PASFIND2
LIS