```
BBBBBBBBBBBB        AAAAAAAAA        SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBBBBBBBBBBB        AAAAAAAAA        SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBBBBBBBBBBB        AAAAAAAAA        SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBB         BBB   AAA         AAA   SSS          RRR         RRR         TTT        LLL
BBB         BBB   AAA         AAA   SSS          RRR         RRR         TTT        LLL
BBB         BBB   AAA         AAA   SSS          RRR         RRR         TTT        LLL
BBB         BBB   AAA         AAA   SSS          RRR         RRR         TTT        LLL
BBB         BBB   AAA         AAA   SSS          RRR         RRR         TTT        LLL
BBB         BBB   AAA         AAA   SSS          RRR         RRR         TTT        LLL
BBBBBBBBBBBB      AAA         AAA     SSSSSSSSS  RRRRRRRRRRR             TTT        LLL
BBBBBBBBBBBB      AAA         AAA     SSSSSSSSS  RRRRRRRRRRR             TTT        LLL
BBBBBBBBBBBB      AAA         AAA     SSSSSSSSS  RRRRRRRRRRR             TTT        LLL
BBB         BBB   AAAAAAAAAAAAAAA           SSS  RRR   RRR               TTT        LLL
BBB         BBB   AAAAAAAAAAAAAAA           SSS  RRR   RRR               TTT        LLL
BBB         BBB   AAAAAAAAAAAAAAA           SSS  RRR   RRR               TTT        LLL
BBB         BBB   AAA         AAA           SSS  RRR      RRR            TTT        LLL
BBB         BBB   AAA         AAA           SSS  RRR   RRR               TTT        LLL
BBB         BBB   AAA         AAA           SSS  RRR      RRR            TTT        LLL
BBBBBBBBBBBB      AAA         AAA   SSSSSSSSSSS     RRR         RRR      TTT        LLLLLLLLLLLLLLL
BBBBBBBBBBBB      AAA         AAA   SSSSSSSSSSS     RRR         RRR      TTT        LLLLLLLLLLLLLLL
BBBBBBBBBBBB      AAA         AAA   SSSSSSSSSSS     RRR         RRR      TTT        LLLLLLLLLLLLLLL
```

```
BBBBBBBB      AAAAAA      SSSSSSSS    SSSSSSSS    IIIIII      GGGGGGGG  NN      NN    AAAAAA    LL
BBBBBBBB      AAAAAA      SSSSSSSS    SSSSSSSS    IIIIII      GGGGGGGG  NN      NN    AAAAAA    LL
BB      BB  AA      AA  SS          SS              II      GG          NN      NN  AA      AA  LL
BB      BB  AA      AA  SS          SS              II      GG          NN      NN  AA      AA  LL
BB      BB  AA      AA  SS          SS              II      GG          NNNN    NN  AA      AA  LL
BB      BB  AA      AA  SS          SS              II      GG          NNNN    NN  AA      AA  LL
BBBBBBBB    AA      AA    SSSSSS      SSSSSS        II      GG          NN  NN  NN  AA      AA  LL
BBBBBBBB    AA      AA    SSSSSS      SSSSSS        II      GG          NN  NN  NN  AA      AA  LL
BB      BB  AAAAAAAAAA          SS          SS      II      GG  GGGGGG  NN    NNNN  AAAAAAAAAA  LL
BB      BB  AAAAAAAAAA          SS          SS      II      GG  GGGGGG  NN    NNNN  AAAAAAAAAA  LL
BB      BB  AA      AA          SS          SS      II      GG      GG  NN      NN  AA      AA  LL                  ....
BB      BB  AA      AA          SS          SS      II      GG      GG  NN      NN  AA      AA  LL                  ....
BBBBBBBB    AA      AA  SSSSSSSS    SSSSSSSS      IIIIII      GGGGGG    NN      NN  AA      AA  LLLLLLLLLL          ....
BBBBBBBB    AA      AA  SSSSSSSS    SSSSSSSS      IIIIII      GGGGGG    NN      NN  AA      AA  LLLLLLLLLL          ....

LL              IIIIII      SSSSSSSS
LL              IIIIII      SSSSSSSS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II        SSSSSS
LL                II        SSSSSS
LL                II              SS
LL                II              SS
LL                II              SS
LL                II              SS
LLLLLLLLLL      IIIIII    SSSSSSSS
LLLLLLLLLL      IIIIII    SSSSSSSS
```

```
;     1          0001  0 MODULE BAS$$SIGNAL_IO (                          !
;     2          0002  0                     IDENT = '1-023'             ! File: BASSIGNAL.B32
;     3          0003  0                     ) =                         ! Edit: MDL1023
;     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:     BASIC-PLUS-2 I/O and Error Handling
;    32          0032  1 !
;    33          0033  1 ! ABSTRACT:
;    34          0034  1 !
;    35          0035  1 !       This module contains BAS$$SIGNAL_IO, which is called following
;    36          0036  1 !       any I/O error.  If requested, it examines the RMS control blocks
;    37          0037  1 !       and signals the proper BASIC error.  Another entry point,
;    38          0038  1 !       BAS$$STOP_IO, guarantees never to return to the caller.
;    39          0039  1 !
;    40          0040  1 ! ENVIRONMENT:  VAX/VMS user mode
;    41          0041  1 !
;    42          0042  1 ! AUTHOR: John Sauter, CREATION DATE: 08-DEC-78
;    43          0043  1 !
;    44          0044  1 ! MODIFIED BY:
;    45          0045  1 !
;    46          0046  1 ! 1-001 - Original.  JBS 08-DEC-78
;    47          0047  1 ! 1-002 - Revise to take CCB via R11 and only need one explicit
;    48          0048  1 !         argument.  JBS 11-DEC-78
;    49          0049  1 ! 1-003 - Compute user PC.  JBS 19-DEC-78
;    50          0050  1 ! 1-004 - Put code in proper PSECT.  JBS 21-DEC-78
;    51          0051  1 ! 1-005 - If this is OPEN and there is no error code in the RAB, extract
;    52          0052  1 !         the error code from the FAB.  JBS 27-DEC-78
;    53          0053  1 ! 1-006 - Change the prefix for BASIC stack frame offsets to BSF$.
;    54          0054  1 !         JBS 08-FEB-1979
;    55          0055  1 ! 1-007 - Don't force the error code to SEVERE by calling LIB$STOP
;    56          0056  1 !         from BAS$SIGNAL_IO.  JBS 20-FEB-1979
;    57          0057  1 ! 1-008 - Use BASIOERR.REQ to define the I/O error codes.  JBS 20-FEB-1979
```

```
;    58          0058  1 ! 1-009 - Take numbers outside of the normal range of error codes
;    59          0059  1 !           to mean BASIC (rather than RMS) errors.  JBS 06-APR-1979
;    60          0060  1 ! 1-010 - Don't use BAS$K_ON_CHAFIL since it does not get defined.
;    61          0061  1 !           JBS 06-APR-1979
;    62          0062  1 ! 1-011 - Don't pass RMS information unless we are asked to compute the
;    63          0063  1 !           BASIC error code from it.  JBS 06-APR-1979
;    64          0064  1 ! 1-012 - Add the error codes for INDEXED I/O.  JBS 09-APR-1979
;    65          0065  1 ! 1-013 - If there is a FAB connected to the RAB, and the RAB does not
;    66          0066  1 !           show an error, get the status from the FAB.  This is for
;    67          0067  1 !           the $EXTEND macro issued by the virtual memory support.
;    68          0068  1 !           JBS 24-MAY-1979
;    69          0069  1 ! 1-014 - If the ISB indicates that this is a to-memory operation,
;    70          0070  1 !           transfer control to BAS$$STOP or BAS$$SIGNAL.
;    71          0071  1 !           JBS 24-MAY-1979
;    72          0072  1 ! 1-015 - Correct an error in the translate table.  JBS 30-JUL-1979
;    73          0073  1 ! 1-016 - Make the SQO error give "Illegal Operation".  JBS 02-AUG-1979
;    74          0074  1 ! 1-017 - Add BAS$$SIGNAL_RMS.  JBS 09-AUG-1979
;    75          0075  1 ! 1-018 - Change BAS$$SIGNAL_RMS to BAS$$STOP_RMS.  Only the comment
;    76          0076  1 !           above was actually wrong.  JBS 22-AUG-1979
;    77          0077  1 ! 1-019 - if 0(fp) is 0 then CALC_USER_PC should bail out before it gets a
;    78          0078  1 !           access violation.  FM 15-MAY-81.
;    79          0079  1 ! 1-020 - Map RMS$_TNS onto BAS$K_LINTOOLON, and RMS$_SPE onto BAS$K_ERRFILCOR.
;    80          0080  1 !           PL 27-Oct-81
;    81          0081  1 ! 1-021 - Remove map of RMS$_LBL, since that status is never returned by RMS.
;    82          0082  1 !           SBL 11-Nov-1982
;    83          0083  1 ! 1-022 - change ERRFILCOR to EXRMSSHR, seeing as we only signal it when the
;    84          0084  1 !           system runs out of RMSSHR.  MDL 5-Jan-1984
;    85          0085  1 ! 1-023 - map RMS$_RNL to BAS$K_NO_CURREC.  MDL 3-May-1984
;    86          0086  1 !--
;    87          0087  1
;    88          0088  1 !<BLF/PAGE>
```

```
;    90          0089  1 !
;    91          0090  1 ! SWITCHES:
;    92          0091  1 !
;    93          0092  1
;    94          0093  1 SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
;    95          0094  1
;    96          0095  1 !
;    97          0096  1 ! LINKAGES:
;    98          0097  1 !
;    99          0098  1
;   100          0099  1 REQUIRE 'RTLIN:OTSLNK';                         ! Define all linkages
;   101          0528  1
;   102          0529  1 !
;   103          0530  1 ! TABLE OF CONTENTS:
;   104          0531  1 !
;   105          0532  1
;   106          0533  1 FORWARD ROUTINE
;   107          0534  1     TRANSLATE_RMS,                              ! Translate an RMS error code
;   108          0535  1     CALC_USER_PC,                               ! Calculate the user's PC
;   109          0536  1     BAS$$SIGNAL_IO : CALL_CCB NOVALUE,          ! Signal an I/O error
;   110          0537  1     BAS$$STOP_IO : CALL_CCB NOVALUE,            ! Signal a fatal I/O error
;   111          0538  1     BAS$$STOP_RMS : NOVALUE;                    ! Signal a fatal I/O error
;   112          0539  1
;   113          0540  1 !
;   114          0541  1 ! INCLUDE FILES:
;   115          0542  1 !
;   116          0543  1
;   117          0544  1 REQUIRE 'RTLIN:RTLPSECT';                       ! Define DECLARE_PSECTS macro
;   118          0639  1
;   119          0640  1 REQUIRE 'RTLML:OTSLUB';                         ! Logical Unit Block definitions
;   120          0780  1
;   121          0781  1 REQUIRE 'RTLML:OTSISB';                         ! I/O Statement Block definitions
;   122          0949  1
;   123          0950  1 REQUIRE 'RTLIN:BASFRAME';                       ! Define BASIC frame structure
;   124          1153  1
;   125          1154  1 REQUIRE 'RTLIN:BASIOERR';                       ! Define I/O error codes.
;   126          1207  1
;   127          1208  1 LIBRARY 'RTLSTARLE';                           ! System Library for RMS symbols
;   128          1209  1
;   129          1210  1 !
;   130          1211  1 ! MACROS:
;   131          1212  1 !
;   132          1213  1 !       NONE
;   133          1214  1 !
;   134          1215  1 ! EQUATED SYMBOLS:
;   135          1216  1 !
;   136          1217  1 !       NONE
;   137          1218  1 !
;   138          1219  1 ! PSECTS:
;   139          1220  1 !
;   140          1221  1 DECLARE_PSECTS (BAS);                          ! Declare PSECTs for BAS facility
;   141          1222  1 !
;   142          1223  1 ! OWN STORAGE:
;   143          1224  1 !
;   144          1225  1 !       NONE
;   145          1226  1 !
;   146          1227  1 ! EXTERNAL REFERENCES:
```

```
;  147          1228  1 !
;  148          1229  1
;  149          1230  1 EXTERNAL ROUTINE
;  150          1231  1     BAS$$STOP : NOVALUE,                    ! Signal a fatal BASIC error
;  151          1232  1     BAS$$SIGNAL : NOVALUE,                  ! Signal a BASIC error
;  152          1233  1     LIB$SIGNAL : NOVALUE,                   ! Signal a non-fatal error
;  153          1234  1     LIB$STOP : NOVALUE,                     ! Signal a fatal error
;  154          1235  1     BAS$HANDLER,                            ! Mark the user's frame
;  155          1236  1     BAS$$COND_VAL;                          ! Make 32-bit error codes
;  156          1237  1
;  157          1238  1 !+
;  158          1239  1 ! The following are the error codes used in this module.
;  159          1240  1 ! For the meanings of these error codes, see the BAS$ERRTXT module.
;  160          1241  1 !-
;  161          1242  1
;  162          1243  1 EXTERNAL LITERAL
;  163          1244  1     BAS$K_BADDIRDEV : UNSIGNED (8),
;  164          1245  1     BAS$K_BADRECIDE : UNSIGNED (8),
;  165          1246  1     BAS$K_BADRECVAL : UNSIGNED (8),
;  166          1247  1     BAS$K_CANFINFIL : UNSIGNED (8),
;  167          1248  1     BAS$K_CANOPEFIL : UNSIGNED (8),
;  168          1249  1     BAS$K_CORFILSTR : UNSIGNED (8),
;  169          1250  1     BAS$K_DEVHUNWRI : UNSIGNED (8),
;  170          1251  1     BAS$K_DIRERR : UNSIGNED (8),
;  171          1252  1     BAS$K_ENDFILDEV : UNSIGNED (8),
;  172          1253  1     BAS$K_FILEXPDAT : UNSIGNED (8),
;  173          1254  1     BAS$K_FATSYSIO_ : UNSIGNED (8),
;  174          1255  1     BAS$K_FILACPFAI : UNSIGNED (8),
;  175          1256  1     BAS$K_FILIS_LOC : UNSIGNED (8),
;  176          1257  1     BAS$_FORFILOSE,
;  177          1258  1     BAS$K_ILLALLCLA : UNSIGNED (8),
;  178          1259  1     BAS$K_ILLFILNAM : UNSIGNED (8),
;  179          1260  1     BAS$K_ILLILLACC : UNSIGNED (8),
;  180          1261  1     BAS$K_ILLOPE : UNSIGNED (8),
;  181          1262  1     BAS$K_ILLRECACC : UNSIGNED (8),
;  182          1263  1     BAS$K_ILLRECFIL : UNSIGNED (8),
;  183          1264  1     BAS$K_ILLRECFOR : UNSIGNED (8),
;  184          1265  1     BAS$K_ILLUSA : UNSIGNED (8),
;  185          1266  1     BAS$K_ILLUSADEV : UNSIGNED (8),
;  186          1267  1     BAS$K_INVFILOPT : UNSIGNED (8),
;  187          1268  1     BAS$K_INVKEYREF : UNSIGNED (8),
;  188          1269  1     BAS$K_INVRFAFIE : UNSIGNED (8),
;  189          1270  1     BAS$K_KEYSIZTOO : UNSIGNED (8),
;  190          1271  1     BAS$K_KEYWAIEXH : UNSIGNED (8),
;  191          1272  1     BAS$K_NAMACCNOW : UNSIGNED (8),
;  192          1273  1     BAS$K_NODNAMERR : UNSIGNED (8),
;  193          1274  1     BAS$K_NOTENDFIL : UNSIGNED (8),
;  194          1275  1     BAS$K_NO_CURREC : UNSIGNED (8),
;  195          1276  1     BAS$K_NO_ROOUSE : UNSIGNED (8),
;  196          1277  1     BAS$_ON_CHAFIL,
;  197          1278  1     BAS$K_PROVIO : UNSIGNED (8),
;  198          1279  1     BAS$K_RECALREXI : UNSIGNED (8),
;  199          1280  1     BAS$K_RECBUCLOC : UNSIGNED (8),
;  200          1281  1     BAS$K_RECFILTOO : UNSIGNED (8),
;  201          1282  1     BAS$K_RECHASBEE : UNSIGNED (8),
;  202          1283  1     BAS$K_RECLOCFAI : UNSIGNED (8),
;  203          1284  1     BAS$K_RECNOTFOU : UNSIGNED (8),
```

```
;  204          1285  1      BAS$K_RECNUMEXC : UNSIGNED (8),
;  205          1286  1      BAS$K_SIZRECINV : UNSIGNED (8),
;  206          1287  1      BAS$K_TAPBOTDET : UNSIGNED (8),
;  207          1288  1      BAS$K_TAPRECNOT : UNSIGNED (8),
;  208          1289  1      BAS$K_KEYNOTCHA : UNSIGNED (8),
;  209          1290  1      BAS$K_DUPKEYDET : UNSIGNED (8),
;  210          1291  1      BAS$K_ILLKEYATT : UNSIGNED (8),
;  211          1292  1      BAS$K_NO_PRIKEY : UNSIGNED (8),
;  212          1293  1      BAS$K_KEYFIEBEY : UNSIGNED (8),
;  213          1294  1      BAS$K_PRIKEYOUT : UNSIGNED (8),
;  214          1295  1      BAS$K_KEYLARTHA : UNSIGNED (8),
;  215          1296  1      BAS$K_INDNOTFUL : UNSIGNED (8),
;  216          1297  1      BAS$K_EXRMSSHR  : UNSIGNED (8),
;  217          1298  1      BAS$K_LINTOOLON : UNSIGNED (8);
;  218          1299  1
```

```
;  220          1300  1 ROUTINE TRANSLATE_RMS (                       ! Translate an RMS error code
;  221          1301  1         STS,                                  ! The RMS STS longword
;  222          1302  1         STV,                                  ! The RMS STV longword
;  223          1303  1         OPEN_FLAG                             ! True if this is an OPEN failure
;  224          1304  1     ) =
;  225          1305  1
;  226          1306  1 !++
;  227          1307  1 ! FUNCTIONAL DESCRIPTION:
;  228          1308  1 !
;  229          1309  1 !       Examines the status values to determine the nature of the RMS
;  230          1310  1 !       error and returns the proper BASIC error code.
;  231          1311  1 !
;  232          1312  1 ! FORMAL PARAMETERS:
;  233          1313  1 !
;  234          1314  1 !       STS.rl.v        The RMS STS field, which contains the major
;  235          1315  1 !                       information about the error.
;  236          1316  1 !       STV.rl.v        The RMS STV field, which contains some extra
;  237          1317  1 !                       error information for some STS values.
;  238          1318  1 !       OPEN_FLAG.rv.v  True (=1) if we are opening a file.
;  239          1319  1 !
;  240          1320  1 ! IMPLICIT INPUTS:
;  241          1321  1 !
;  242          1322  1 !       NONE
;  243          1323  1 !
;  244          1324  1 ! IMPLICIT OUTPUTS:
;  245          1325  1 !
;  246          1326  1 !       NONE
;  247          1327  1 !
;  248          1328  1 ! ROUTINE VALUE:
;  249          1329  1 !
;  250          1330  1 !       The 32-bit BASIC error message code corresponding to the
;  251          1331  1 !       RMS error.
;  252          1332  1 !
;  253          1333  1 ! SIDE EFFECTS:
;  254          1334  1 !
;  255          1335  1 !       NONE
;  256          1336  1 !
;  257          1337  1 !--
;  258          1338  1
;  259          1339  2     BEGIN
;  260          1340  2
;  261          1341  2     LOCAL
;  262          1342  2         BASIC_ERR_CODE;                         ! The 32-bit BASIC error code
;  263          1343  2
;  264          1344  2 !+
;  265          1345  2 ! The following SELECTONE statement searches the translation table
;  266          1346  2 ! for the BASIC error code corresponding to the STS value passed.
;  267          1347  2 !-
;  268          1348  3     BASIC_ERR_CODE = (SELECTONE (.STS) OF
;  269          1349  3         SET
;  270          1350  3         [RMS$_ANI] : BAS$K_TAPRECNOT;
;  271          1351  3         [RMS$_ATR] : BAS$K_FILACPFAI;
;  272          1352  3         [RMS$_ATW] : BAS$K_FILACPFAI;
;  273          1353  3         [RMS$_BOF] : BAS$K_TAPBOTDET;
;  274          1354  3         [RMS$_CHG] : BAS$K_KEYNOTCHA;
;  275          1355  3         [RMS$_CHK] : BAS$K_CORFILSTR;
;  276          1356  3         [RMS$_CUR] : BAS$K_NO_CURREC;
```

```
;  277          1357  3          [RMS$_DAC] : BAS$K_FILACPFAI;
;  278          1358  3          [RMS$_DEL] : BAS$K_RECHASBEE;
;  279          1359  3          [RMS$_DEV] : BAS$K_ILLUSADEV;
;  280          1360  3          [RMS$_DIR] : BAS$K_BADDIRDEV;
;  281          1361  3          [RMS$_DNF] : BAS$K_BADDIRDEV;
;  282          1362  3          [RMS$_DNR] : BAS$K_DEVHUNWRI;
;  283          1363  3          [RMS$_DPE] : BAS$K_FILACPFAI;
;  284          1364  3          [RMS$_DUP] : BAS$K_DUPKEYDET;
;  285          1365  3          [RMS$_ENV] : BAS$K_ILLUSA;
;  286          1366  3          [RMS$_EOF] : BAS$K_ENDFILDEV;
;  287          1367  3          [RMS$_EXP] : BAS$K_FILEXPDAT;          ! New with BASIC-PLUS-2/VAX
;  288          1368  3          [RMS$_EXT] : BAS$K_FILACPFAI;
;  289          1369  3          [RMS$_FAC] : BAS$K_ILLILLACC;
;  290          1370  3          [RMS$_FEX] : BAS$K_NAMACCNOW;
;  291          1371  3          [RMS$_FLG] : BAS$K_ILLKEYATT;
;  292          1372  3          [RMS$_FLK] : BAS$K_FILIS_LOC;
;  293          1373  3          [RMS$_FNF] : BAS$K_CANFINFIL;
;  294          1374  3          [RMS$_FNM] : BAS$K_ILLFILNAM;
;  295          1375  3          [RMS$_FOP] : BAS$K_INVFILOPT;
;  296          1376  3          [RMS$_FUL] : BAS$K_NO_ROOUSE;
;  297          1377  3          [RMS$_IOP] : BAS$K_ILLOPE;
;  298          1378  3          [RMS$_IRC] : BAS$K_ILLRECFIL;
;  299          1379  3          [RMS$_KBF] : BAS$K_BADRECIDE;
;  300          1380  3          [RMS$_KEY] : BAS$K_BADRECIDE;
;  301          1381  3          [RMS$_KRF] : BAS$K_INVKEYREF;
;  302          1382  3          [RMS$_KSZ] : BAS$K_KEYSIZTOO;
;  303          1383  3          [RMS$_MKD] : BAS$K_FILACPFAI;
;  304          1384  3          [RMS$_MRN] : BAS$K_RECNUMEXC;
;  305          1385  3          [RMS$_MRS] : BAS$K_BADRECVAL;
;  306          1386  3          [RMS$_NEF] : BAS$K_NOTENDFIL;
;  307          1387  3          [RMS$_NOD] : BAS$K_NODNAMERR;          ! New with BASIC-PLUS-2/VAX
;  308          1388  3          [RMS$_NPK] : BAS$K_NO_PRIKEY;
;  309          1389  3          [RMS$_OK_IDX] : BAS$K_INDNOTFUL;
;  310          1390  3          [RMS$_OK_RLK] : BAS$K_RECLOCFAI;
;  311          1391  3          [RMS$_PLG] : BAS$K_CORFILSTR;
;  312          1392  3          [RMS$_POS] : BAS$K_KEYFIEBEY;
;  313          1393  3          [RMS$_PRV] : BAS$K_PROVIO;
;  314          1394  3          [RMS$_RAC] : BAS$K_ILLRECACC;
;  315          1395  3          [RMS$_RAT] : BAS$K_ILLRECACC;
;  316          1396  3          [RMS$_RBF] : BAS$K_BADRECIDE;
;  317          1397  3          [RMS$_RER] : BAS$K_FILACPFAI;
;  318          1398  3          [RMS$_REX] : BAS$K_RECALREXI;
;  319          1399  3          [RMS$_RFA] : BAS$K_INVRFAFIE;
;  320          1400  3          [RMS$_RFM] : BAS$K_ILLRECFOR;
;  321          1401  3          [RMS$_RLK] : BAS$K_RECBUCLOC;
;  322          1402  3          [RMS$_RMV] : BAS$K_FILACPFAI;
;  323          1403  3          [RMS$_RNF] : BAS$K_RECNOTFOU;
;  324          1404  3          [RMS$_RNL] : BAS$K_NO_CURREC;
;  325          1405  3          [RMS$_RPL] : BAS$K_FILACPFAI;
;  326          1406  3          [RMS$_RRV] : BAS$K_CORFILSTR;
;  327          1407  3          [RMS$_RSZ] : BAS$K_SIZRECINV;
;  328          1408  3          [RMS$_RTB] : BAS$K_RECFILTOO;
;  329          1409  3          [RMS$_SEQ] : BAS$K_PRIKEYOUT;
;  330          1410  3          [RMS$_SHR] : BAS$K_ILLALLCLA;
;  331          1411  3          [RMS$_SIZ] : BAS$K_KEYLARTHA;
;  332          1412  3          [RMS$_SPE] : BAS$K_EXRMSSHR;
;  333          1413  3          [RMS$_SQO] : BAS$K_ILLOPE;
```

```
;  334          1414  3          [RMS$_SYN] : BAS$K_ILLFILNAM;
;  335          1415  3          [RMS$_SYS] : BAS$K_DIRERR;
;  336          1416  3          [RMS$_TMO] : BAS$K_KEYWAIEXH;
;  337          1417  3          [RMS$_TNS] : BAS$K_LINTOOLON;
;  338          1418  3          [RMS$_TRE] : BAS$K_CORFILSTR;
;  339          1419  3          [RMS$_TYP] : BAS$K_ILLFILNAM;
;  340          1420  3          [RMS$_VER] : BAS$K_ILLFILNAM;
;  341          1421  3          [RMS$_WBE] : BAS$K_FILACPFAI;
;  342          1422  3          [RMS$_WER] : BAS$K_FILACPFAI;
;  343          1423  3          [RMS$_WLK] : BAS$K_DEVHUNWRI;
;  344          1424  3          [RMS$_WPL] : BAS$K_FILACPFAI;
;  345          1425  3          [OTHERWISE] : 0;
;  346          1426  2          TES);
;  347          1427  2
;  348          1428  3      IF (.BASIC_ERR_CODE EQL 0)
;  349          1429  2      THEN
;  350          1430  2  !+
;  351          1431  2  ! The code is not in the above table.  If we are opening a file give
;  352          1432  2  ! the message "Can't open file", otherwise "Fatal system I/O error".
;  353          1433  2  !-
;  354          1434  2
;  355          1435  2          IF (.OPEN_FLAG) THEN BASIC_ERR_CODE = BAS$K_CANOPEFIL ELSE BASIC_ERR_CODE = BAS$K_FATSYSIO_;
;  356          1436  2
;  357          1437  2      RETURN (.BASIC_ERR_CODE);
;  358          1438  1      END;                                    ! end of TRANSLATE_RMS


                                .TITLE  BAS$$SIGNAL_IO
                                .IDENT  \1-023\

                                .EXTRN  BAS$$STOP, BAS$$SIGNAL
                                .EXTRN  LIB$SIGNAL, LIB$STOP
                                .EXTRN  BAS$HANDLER, BAS$$COND_VAL
                                .EXTRN  BAS$K_BADDIRDEV
                                .EXTRN  BAS$K_BADRECIDE
                                .EXTRN  BAS$K_BADRECVAL
                                .EXTRN  BAS$K_CANFINFIL
                                .EXTRN  BAS$K_CANOPEFIL
                                .EXTRN  BAS$K_CORFILSTR
                                .EXTRN  BAS$K_DEVHUNWRI
                                .EXTRN  BAS$K_DIRERR, BAS$K_ENDFILDEV
                                .EXTRN  BAS$K_FILEXPDAT
                                .EXTRN  BAS$K_FATSYSIO
                                .EXTRN  BAS$K_FILACPFAI
                                .EXTRN  BAS$K_FILIS_LOC
                                .EXTRN  BAS$_FORFILUSE, BAS$K_ILLALLCLA
                                .EXTRN  BAS$K_ILLFILNAM
                                .EXTRN  BAS$K_ILLILLACC
                                .EXTRN  BAS$K_ILLOPE, BAS$K_ILLRECACC
                                .EXTRN  BAS$K_ILLRECFIL
                                .EXTRN  BAS$K_ILLRECFOR
                                .EXTRN  BAS$K_ILLUSA, BAS$K_ILLUSADEV
                                .EXTRN  BAS$K_INVFILOPT
                                .EXTRN  BAS$K_INVKEYREF
                                .EXTRN  BAS$K_INVRFAFIE
                                .EXTRN  BAS$K_KEYSIZTOO
                                .EXTRN  BAS$K_KEYWAIEXH
```

```
                        .EXTRN  BAS$K_NAMACCNOW
                        .EXTRN  BAS$K_NODNAMERR
                        .EXTRN  BAS$K_NOTENDFIL
                        .EXTRN  BAS$K_NO_CURREC
                        .EXTRN  BAS$K_NO_ROOUSE
                        .EXTRN  BAS$_ON_CHAFIL, BAS$K_PROVIO
                        .EXTRN  BAS$K_RECALREXI
                        .EXTRN  BAS$K_RECBUCLOC
                        .EXTRN  BAS$K_RECFILTOO
                        .EXTRN  BAS$K_RECHASBEE
                        .EXTRN  BAS$K_RECLOCFAI
                        .EXTRN  BAS$K_RECNOTFOU
                        .EXTRN  BAS$K_RECNUMEXC
                        .EXTRN  BAS$K_SIZRECINV
                        .EXTRN  BAS$K_TAPBOTDET
                        .EXTRN  BAS$K_TAPRECNOT
                        .EXTRN  BAS$K_KEYNOTCHA
                        .EXTRN  BAS$K_DUPKEYDET
                        .EXTRN  BAS$K_ILLKEYATT
                        .EXTRN  BAS$K_NO_PRIKEY
                        .EXTRN  BAS$K_KEYFIEBEY
                        .EXTRN  BAS$K_PRIKEYOUT
                        .EXTRN  BAS$K_KEYLARTHA
                        .EXTRN  BAS$K_INDNOTFUL
                        .EXTRN  BAS$K_EXRMSSHR, BAS$K_LINTOOLON

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

                0000 00000 TRANSLATE_RMS:
                                .WORD   Save nothing                            ; 1300
                50      04  AC  D0 00002       MOVL    STS, R0                ; 1348
0001840C        8F          50  D1 00006       CMPL    R0, #99340             ; 1350
                            06  12 0000D       BNEQ    1$
                50      00G 8F  9A 0000F       MOVZBL  #BAS$K_TAPRECNOT, BASIC_ERR_CODE
                            6D  11 00013       BRB     8$
0001C0CC        8F          50  D1 00015 1$:   CMPL    R0, #114892            ; 1351
                            46  13 0001C       BEQL    6$
0001C0D4        8F          50  D1 0001E       CMPL    R0, #114900            ; 1352
                            3D  13 00025       BEQL    6$
00018198        8F          50  D1 00027       CMPL    R0, #98712             ; 1353
                            06  12 0002E       BNEQ    2$
                50      00G 8F  9A 00030       MOVZBL  #BAS$K_TAPBOTDET, BASIC_ERR_CODE
                            65  11 00034       BRB     11$
0001849C        8F          50  D1 00036 2$:   CMPL    R0, #99484             ; 1354
                            06  12 0003D       BNEQ    3$
                50      00G 8F  9A 0003F       MOVZBL  #BAS$K_KEYNOTCHA, BASIC_ERR_CODE
                            7A  11 00043       BRB     15$
000184A4        8F          50  D1 00045 3$:   CMPL    R0, #99492             ; 1355
                            03  12 0004C       BNEQ    4$
                        036D    31 0004E       BRW     103$
000184B4        8F          50  D1 00051 4$:   CMPL    R0, #99508             ; 1356
                            03  12 00058       BNEQ    5$
                        029B    31 0005A       BRW     78$
0001C012        8F          50  D1 0005D 5$:   CMPL    R0, #114706            ; 1357
                            4A  13 00064 6$:   BEQL    14$
00018262        8F          50  D1 00066       CMPL    R0, #98914             ; 1358
                            06  12 0006D       BNEQ    7$
```

```
                50      00G  8F  9A 0006F          MOVZBL  #BAS$K_RECHASBEE, BASIC_ERR_CODE
                             78  11 00073          BRB     20$
000184C4        8F           50  D1 00075 7$:      CMPL    R0, #99524                      ;1359
                             07  12 0007C          BNEQ    9$
                50      00G  8F  9A 0007E          MOVZBL  #BAS$K_ILLUSADEV, BASIC_ERR_CODE
                           0084  31 00082 8$:       BRW     24$
000184CC        8F           50  D1 00085 9$:      CMPL    R0, #99532                      ;1360
                             09  13 0008C          BEQL    10$
0001C04A        8F           50  D1 0008E          CMPL    R0, #114762                     ;1361
                             06  12 00095          BNEQ    12$
                50      00G  8F  9A 00097 10$:     MOVZBL  #BAS$K_BADDIRDEV, BASIC_ERR_CODE
                             7B  11 0009B 11$:     BRB     26$
00018272        8F           50  D1 0009D 12$:     CMPL    R0, #98930                      ;1362
                             03  12 000A4          BNEQ    13$
                           034E  31 000A6          BRW     109$
0001C03A        8F           50  D1 000A9 13$:     CMPL    R0, #114746                     ;1363
                             45  13 000B0 14$:     BEQL    22$
000184EC        8F           50  D1 000B2          CMPL    R0, #99564                      ;1364
                             06  12 000B9          BNEQ    16$
                50      00G  8F  9A 000BB          MOVZBL  #BAS$K_DUPKEYDET, BASIC_ERR_CODE
                             0D  11 000BF 15$:     BRB     17$
00018724        8F           50  D1 000C1 16$:     CMPL    R0, #100132                     ;1365
                             06  12 000C8          BNEQ    18$
                50      00G  8F  9A 000CA          MOVZBL  #BAS$K_ILLUSA, BASIC_ERR_CODE
                             76  11 000CE 17$:     BRB     30$
0001827A        8F           50  D1 000D0 18$:     CMPL    R0, #98938                      ;1366
                             07  12 000D7          BNEQ    19$
                50      00G  8F  9A 000D9          MOVZBL  #BAS$K_ENDFILDEV, BASIC_ERR_CODE
                           0081  31 000DD          BRW     33$
000182C2        8F           50  D1 000E0 19$:     CMPL    R0, #99010                      ;1367
                             07  12 000E7          BNEQ    21$
                50      00G  8F  9A 000E9          MOVZBL  #BAS$K_FILEXPDAT, BASIC_ERR_CODE
                           0081  31 000ED 20$:     BRW     35$
0001C022        8F           50  D1 000F0 21$:     CMPL    R0, #114722                     ;1368
                             03  12 000F7 22$:     BNEQ    23$
                           030A  31 000F9          BRW     112$
00018514        8F           50  D1 000FC 23$:     CMPL    R0, #99604                      ;1369
                             06  12 00103          BNEQ    25$
                50      00G  8F  9A 00105          MOVZBL  #BAS$K_ILLILLACC, BASIC_ERR_CODE
                             0D  11 00109 24$:     BRB     26$
00018282        8F           50  D1 0010B 25$:     CMPL    R0, #98946                      ;1370
                             06  12 00112          BNEQ    27$
                50      00G  8F  9A 00114          MOVZBL  #BAS$K_NAMACCNOW, BASIC_ERR_CODE
                             73  11 00118 26$:     BRB     38$
0001851C        8F           50  D1 0011A 27$:     CMPL    R0, #99612                      ;1371
                             07  12 00121          BNEQ    28$
                50      00G  8F  9A 00123          MOVZBL  #BAS$K_ILLKEYATT, BASIC_ERR_CODE
                           0087  31 00127          BRW     42$
0001828A        8F           50  D1 0012A 28$:     CMPL    R0, #98954                      ;1372
                             06  12 00131          BNEQ    29$
                50      00G  8F  9A 00133          MOVZBL  #BAS$K_FILIS_LOC, BASIC_ERR_CODE
                             0D  11 00137          BRB     30$
00018292        8F           50  D1 00139 29$:     CMPL    R0, #98962                      ;1373
                             06  12 00140          BNEQ    31$
                50      00G  8F  9A 00142          MOVZBL  #BAS$K_CANFINFIL, BASIC_ERR_CODE
                             78  11 00146 30$:     BRB     44$
0001852C        8F           50  D1 00148 31$:     CMPL    R0, #99628                      ;1374
```

```
                        03  12  0014F          BNEQ    32$
                      0282  31  00151          BRW     106$
0001853C  8F            50  D1  00154 32$:     CMPL    R0, #99644                          ;1375
                        07  12  0015B          BNEQ    34$
                50  00G  8F  9A  0015D          MOVZBL  #BAS$K_INVFILOPT, BASIC_ERR_CODE
                      0086  31  00161 33$:     BRW     48$
00018544  8F            50  D1  00164 34$:     CMPL    R0, #99652                          ;1376
                        07  12  0016B          BNEQ    36$
                50  00G  8F  9A  0016D          MOVZBL  #BAS$K_NO_ROOUSE, BASIC_ERR_CODE
                      0085  31  00171 35$:     BRW     50$
00018574  8F            50  D1  00174 36$:     CMPL    R0, #99700                          ;1377
                        03  12  0017B          BNEQ    37$
                      01F9  31  0017D          BRW     95$
0001857C  8F            50  D1  00180 37$:     CMPL    R0, #99708                          ;1378
                        06  12  00187          BNEQ    39$
                50  00G  8F  9A  00189          MOVZBL  #BAS$K_ILLRECFIL, BASIC_ERR_CODE
                        7A  11  0018D 38$:     BRB     52$
0001858C  8F            50  D1  0018F 39$:     CMPL    R0, #99724                          ;1379
                        07  13  00196          BEQL    40$
00018594  8F            50  D1  00198          CMPL    R0, #99732                          ;1380
                        03  12  0019F 40$:     BNEQ    41$
                      00E4  31  001A1          BRW     65$
0001859C  8F            50  D1  001A4 41$:     CMPL    R0, #99740                          ;1381
                        06  12  001AB          BNEQ    43$
                50  00G  8F  9A  001AD          MOVZBL  #BAS$K_INVKEYREF, BASIC_ERR_CODE
                        0D  11  001B1 42$:     BRB     44$
000185A4  8F            50  D1  001B3 43$:     CMPL    R0, #99748                          ;1382
                        06  12  001BA          BNEQ    45$
                50  00G  8F  9A  001BC          MOVZBL  #BAS$K_KEYSIZTOO, BASIC_ERR_CODE
                        77  11  001C0 44$:     BRB     56$
0001C032  8F            50  D1  001C2 45$:     CMPL    R0, #114738                         ;1383
                        03  12  001C9          BNEQ    46$
                      0238  31  001CB          BRW     112$
000185CC  8F            50  D1  001CE 46$:     CMPL    R0, #99788                          ;1384
                        06  12  001D5          BNEQ    47$
                50  00G  8F  9A  001D7          MOVZBL  #BAS$K_RECNUMEXC, BASIC_ERR_CODE
                        0D  11  001DB          BRB     48$
000185D4  8F            50  D1  001DD 47$:     CMPL    R0, #99796                          ;1385
                        06  12  001E4          BNEQ    49$
                50  00G  8F  9A  001E6          MOVZBL  #BAS$K_BADRECVAL, BASIC_ERR_CODE
                        78  11  001EA 48$:     BRB     60$
000185E4  8F            50  D1  001EC 49$:     CMPL    R0, #99812                          ;1386
                        07  12  001F3          BNEQ    51$
                50  00G  8F  9A  001F5          MOVZBL  #BAS$K_NOTENDFIL, BASIC_ERR_CODE
                      0081  31  001F9 50$:     BRW     63$
000185F4  8F            50  D1  001FC 51$:     CMPL    R0, #99828                          ;1387
                        07  12  00203          BNEQ    53$
                50  00G  8F  9A  00205          MOVZBL  #BAS$K_NODNAMERR, BASIC_ERR_CODE
                      0080  31  00209 52$:     BRW     66$
000185FC  8F            50  D1  0020C 53$:     CMPL    R0, #99836                          ;1388
                        07  12  00213          BNEQ    54$
                50  00G  8F  9A  00215          MOVZBL  #BAS$K_NO_PRIKEY, BASIC_ERR_CODE
                      0088  31  00219          BRW     68$
00018019  8F            50  D1  0021C 54$:     CMPL    R0, #98329                          ;1389
                        07  12  00223          BNEQ    55$
                50  00G  8F  9A  00225          MOVZBL  #BAS$K_INDNOTFUL, BASIC_ERR_CODE
                      0088  31  00229          BRW     70$
```

```
00018021  8F      50  D1 0022C 55$:   CMPL    R0, #98337                              ; 1390
                  07  12 00233       BNEQ    57$
          50   00G 8F  9A 00235       MOVZBL  #BAS$K_RECLOCFAI, BASIC_ERR_CODE
                0088  31 00239 56$:   BRW     72$
0001861C  8F      50  D1 0023C 57$:   CMPL    R0, #99868                              ; 1391
                  03  12 00243       BNEQ    58$
                0176  31 00245       BRW     103$
00018624  8F      50  D1 00248 58$:   CMPL    R0, #99876                              ; 1392
                  06  12 0024F       BNEQ    59$
          50   00G 8F  9A 00251       MOVZBL  #BAS$K_KEYFIEBEY, BASIC_ERR_CODE
                  7D  11 00255       BRB     74$
0001829A  8F      50  D1 00257 59$:   CMPL    R0, #98970                              ; 1393
                  07  12 0025E       BNEQ    61$
          50   00G 8F  9A 00260       MOVZBL  #BAS$K_PROVIO, BASIC_ERR_CODE
                0086  31 00264 60$:   BRW     76$
00018644  8F      50  D1 00267 61$:   CMPL    R0, #99908                              ; 1394
                  09  13 0026E       BEQL    62$
0001864C  8F      50  D1 00270       CMPL    R0, #99916                              ; 1395
                  06  12 00277       BNEQ    64$
          50   00G 8F  9A 00279 62$:   MOVZBL  #BAS$K_ILLRECACC, BASIC_ERR_CODE
                  7D  11 0027D 63$:   BRB     79$
00018654  8F      50  D1 0027F 64$:   CMPL    R0, #99924                              ; 1396
                  06  12 00286       BNEQ    67$
          50   00G 8F  9A 00288 65$:   MOVZBL  #BAS$K_BADRECIDE, BASIC_ERR_CODE
                  6E  11 0028C 66$:   BRB     79$
0001C0F4  8F      50  D1 0029E 67$:   CMPL    R0, #114932                             ; 1397
                  6E  13 00295       BEQL    81$
000182A2  8F      50  D1 00297       CMPL    R0, #98978                              ; 1398
                  07  12 0029E       BNEQ    69$
          50   00G 8F  9A 002A0       MOVZBL  #BAS$K_RECALREXI, BASIC_ERR_CODE
                008B  31 002A4 68$:   BRW     85$
0001865C  8F      50  D1 002A7 69$:   CMPL    R0, #99932                              ; 1399
                  07  12 002AE       BNEQ    71$
          50   00G 8F  9A 002B0       MOVZBL  #BAS$K_INVRFAFIE, BASIC_ERR_CODE
                008A  31 002B4 70$:   BRW     87$
00018664  8F      50  D1 002B7 71$:   CMPL    R0, #99940                              ; 1400
                  07  12 002BE       BNEQ    73$
          50   00G 8F  9A 002C0       MOVZBL  #BAS$K_ILLRECFOR, BASIC_ERR_CODE
                0089  31 002C4 72$:   BRW     89$
000182AA  8F      50  D1 002C7 73$:   CMPL    R0, #98986                              ; 1401
                  07  12 002CE       BNEQ    75$
          50   00G 8F  9A 002D0       MOVZBL  #BAS$K_RECBUCLOC, BASIC_ERR_CODE
                0088  31 002D4 74$:   BRW     91$
0001C0FC  8F      50  D1 002D7 75$:   CMPL    R0, #114940                             ; 1402
                  25  13 002DE       BEQL    81$
000182B2  8F      50  D1 002E0       CMPL    R0, #98994                              ; 1403
                  06  12 002E7       BNEQ    77$
          50   00G 8F  9A 002E9       MOVZBL  #BAS$K_RECNOTFOU, BASIC_ERR_CODE
                  7F  11 002ED 76$:   BRB     93$
000181A0  8F      50  D1 002EF 77$:   CMPL    R0, #98720                              ; 1404
                  06  12 002F6       BNEQ    80$
          50   00G 8F  9A 002F8 78$:   MOVZBL  #BAS$K_NO_CURREC, BASIC_ERR_CODE
                  7F  11 002FC 79$:   BRB     96$
0001C104  8F      50  D1 002FE 80$:   CMPL    R0, #114948                             ; 1405
                  03  12 00305 81$:   BNEQ    82$
                00FC  31 00307       BRW     112$
00018684  8F      50  D1 0030A 82$:   CMPL    R0, #99972                              ; 1406
```

```
                                03  12 00311          BNEQ    83$                                     ;
                              00A8  31 00313          BRW     103$                                    ;
000186A4    8F                  50  D1 00316 83$:     CMPL    R0, #100004                             ; 1407
                                06  12 0031D          BNEQ    84$                                     ;
                50      00G     8F  9A 0031F          MOVZBL  #BAS$K_SIZRECINV, BASIC_ERR_CODE        ;
                                7F  11 00323          BRB     99$                                     ;
000181A8    8F                  50  D1 00325 84$:     CMPL    R0, #98728                              ; 1408
                                06  12 0032C          BNEQ    86$                                     ;
                50      00G     8F  9A 0032E          MOVZBL  #BAS$K_RECFILTOO, BASIC_ERR_CODE        ;
                                7F  11 00332 85$:     BRB     101$                                    ;
000186AC    8F                  50  D1 00334 86$:     CMPL    R0, #100012                             ; 1409
                                06  12 0033B          BNEQ    88$                                     ;
                50      00G     8F  9A 0033D          MOVZBL  #BAS$K_PKIKEYOUT, BASIC_ERR_CODE        ;
                                7F  11 00341 87$:     BRB     104$                                    ;
000186B4    8F                  50  D1 00343 88$:     CMPL    R0, #100020                             ; 1410
                                06  12 0034A          BNEQ    90$                                     ;
                50      00G     8F  9A 0034C          MOVZBL  #BAS$K_ILLALLCLA, BASIC_ERR_CODE        ;
                                1C  11 00350 89$:     BRB     93$                                     ;
000186BC    8F                  50  D1 00352 90$:     CMPL    R0, #100028                             ; 1411
                                06  12 00359          BNEQ    92$                                     ;
                50      00G     8F  9A 0035B          MOVZBL  #BAS$K_KEYLARTHA, BASIC_ERR_CODE        ;
                                79  11 0035F 91$:     BRB     107$                                    ;
000187A4    8F                  50  D1 00361 92$:     CMPL    R0, #100260                             ; 1412
                                06  12 00368          BNEQ    94$                                     ;
                50      00G     8F  9A 0036A          MOVZBL  #BAS$K_EXRMSSHR, BASIC_ERR_CODE         ;
                                6A  11 0036E 93$:     BRB     107$                                    ;
000186C4    8F                  50  D1 00370 94$:     CMPL    R0, #100036                             ; 1413
                                06  12 00377          BNEQ    97$                                     ;
                50      00G     8F  9A 00379 95$:     MOVZBL  #BAS$K_ILLOPE, BASIC_ERR_CODE           ;
                                7C  11 0037D 96$:     BRB     110$                                    ;
000186D4    8F                  50  D1 0037F 97$:     CMPL    R0, #100052                             ; 1414
                                4E  13 00386          BEQL    106$                                    ;
0001C10C    8F                  50  D1 00388          CMPL    R0, #114956                             ; 1415
                                06  12 0038F          BNEQ    98$                                     ;
                50      00G     8F  9A 00391          MOVZBL  #BAS$K_DIRERR, BASIC_ERR_CODE           ;
                                77  11 00395          BRB     114$                                    ;
000181B0    8F                  50  D1 00397 98$:     CMPL    R0, #98736                              ; 1416
                                06  12 0039E          BNEQ    100$                                    ;
                50      00G     8F  9A 003A0          MOVZBL  #BAS$K_KEYWAIEXH, BASIC_ERR_CODE        ;
                                68  11 003A4 99$:     BRB     114$                                    ;
000181B8    8F                  50  D1 003A6 100$:    CMPL    R0, #98744                              ; 1417
                                06  12 003AD          BNEQ    102$                                    ;
                50      00G     8F  9A 003AF          MOVZBL  #BAS$K_LINTOOLON, BASIC_ERR_CODE        ;
                                59  11 003B3 101$:    BRB     114$                                    ;
000186DC    8F                  50  D1 003B5 102$:    CMPL    R0, #100060                             ; 1418
                                06  12 003BC          BNEQ    105$                                    ;
                50      00G     8F  9A 003BE 103$:    MOVZBL  #BAS$K_CORFILSTR, BASIC_ERR_CODE        ;
                                4A  11 003C2 104$:    BRB     114$                                    ;
000186E4    8F                  50  D1 003C4 105$:    CMPL    R0, #100068                             ; 1419
                                09  13 003CB          BEQL    106$                                    ;
000186FC    8F                  50  D1 003CD          CMPL    R0, #100092                             ; 1420
                                06  12 003D4          BNEQ    108$                                    ;
                50      00G     8F  9A 003D6 106$:    MOVZBL  #BAS$K_ILLFILNAM, BASIC_ERR_CODE        ;
                                32  11 003DA 107$:    BRB     114$                                    ;
0001C12C    8F                  50  D1 003DC 108$:    CMPL    R0, #114988                             ; 1421
                                21  13 003E3          BEQL    112$                                    ;
0001C114    8F                  50  D1 003E5          CMPL    R0, #114964                             ; 1422
```

```
                                 18  13 003EC          BEQL    112$                                     ;
        000182BA  8F                 50  D1 003EE          CMPL    R0, #99002                               ; 1423
                                     06  12 003F5          BNEQ    111$                                     ;
                  50         00G  8F  9A 003F7 109$:   MOVZBL  #BAS$K_DEVHUNWRI, BASIC_ERR_CODE         ;
                                     11  11 003FB 110$:   BRB     114$                                     ;
        0001C11C  8F                 50  D1 003FD 111$:   CMPL    R0, #114972                              ; 1424
                                     06  12 00404          BNEQ    113$                                     ;
                  50         00G  8F  9A 00406 112$:   MOVZBL  #BAS$K_FILACPFAI, BASIC_ERR_CODE         ;
                                     02  11 0040A          BRB     114$                                     ;
                                     50  D4 0040C 113$:   CLRL    BASIC_ERR_CODE                           ; 1425
                                     0D  12 0040E 114$:   BNEQ    116$                                     ; 1428
                  05         0C   AC  E9 00410          BLBC    OPEN_FLAG, 115$                          ; 1435
                  50         00G  8F  9A 00414          MOVZBL  #BAS$K_CANOPEFIL, BASIC_ERR_CODE         ;
                                     04 00418          RET                                              ;
                  50         00G  8F  9A 00419 115$:   MOVZBL  #BAS$K_FATSYSIO_, BASIC_ERR_CODE         ;
                                     04 0041D 116$:   RET                                              ; 1438

; Routine Size:  1054 bytes,    Routine Base:  _BAS$CODE + 0000
```

```
;  360          1439  1 ROUTINE CALC_USER_PC =                        ! Calculate the user's PC
;  361          1440  1
;  362          1441  1 !++
;  363          1442  1 ! FUNCTIONAL DESCRIPTION:
;  364          1443  1 !
;  365          1444  1 !       Search back through the stack to find the user's PC, and return
;  366          1445  1 !       it.
;  367          1446  1 !
;  368          1447  1 ! FORMAL PARAMETERS:
;  369          1448  1 !
;  370          1449  1 !       NONE
;  371          1450  1 !
;  372          1451  1 ! IMPLICIT INPUTS:
;  373          1452  1 !
;  374          1453  1 !       The stack, which holds the process history to this point.
;  375          1454  1 !
;  376          1455  1 ! IMPLICIT OUTPUTS:
;  377          1456  1 !
;  378          1457  1 !       NONE
;  379          1458  1 !
;  380          1459  1 ! ROUTINE VALUE:
;  381          1460  1 !
;  382          1461  1 !       The user's PC, or 0 if no user PC can be found.
;  383          1462  1 !
;  384          1463  1 ! SIDE EFFECTS:
;  385          1464  1 !
;  386          1465  1 !       NONE
;  387          1466  1 !
;  388          1467  1 !--
;  389          1468  1
;  390          1469  2     BEGIN
;  391          1470  2
;  392          1471  2     BUILTIN
;  393          1472  2         FP;
;  394          1473  2
;  395          1474  2     LOCAL
;  396          1475  2         FMP : REF BLOCK [0, BYTE] FIELD (BSF$FCD),      ! Frame under consideration
;  397          1476  2         PREV_FMP : REF BLOCK [0, BYTE] FIELD (BSF$FCD), ! Its predecessor
;  398          1477  2         USER_PC,                                        ! User PC, found in PREV_FMP
;  399          1478  2         SEARCH_COUNTER,                                 ! Prevents the search from running forever
;  400          1479  2         SEARCH_DONE;                                    ! Flags that the search is complete
;  401          1480  2
;  402          1481  2 !+
;  403          1482  2 ! Go back through the stack frames, starting with this one, to find
;  404          1483  2 ! one whose handler is BAS$HANDLER.  The PC stored by its call is the
;  405          1484  2 ! user PC.
;  406          1485  2 !-
;  407          1486  2     SEARCH_DONE = 0;
;  408          1487  2     SEARCH_COUNTER = 0;
;  409          1488  2     FMP = .FP;
;  410          1489  2
;  411          1490  2     WHILE (.SEARCH_DONE EQL 0) DO
;  412          1491  3         BEGIN
;  413          1492  3         PREV_FMP = .FMP;                            ! Remember previous frame
;  414          1493  3         FMP = .PREV_FMP [BSF$A_SAVED_FP];           ! Point to this frame
;  415          1494  3         SEARCH_COUNTER = .SEARCH_COUNTER + 1;       ! We are one level deeper
;  416          1495  3
```

```
;  417          1496  3          IF (.SEARCH_COUNTER GTR 65535) OR .FMP EQLA 0
;  418          1497  3          THEN
;  419          1498  4              BEGIN
;  420          1499  4          !+
;  421          1500  4          ! We have searched too far, or 0(fp) is 0.
;  422          1501  4          ! The stack is probably messed up, or a non-BASIC fram.  Return
;  423          1502  4          ! a zero, and the traceback will show the user the mess.
;  424          1503  4          !-
;  425          1504  4              USER_PC = 0;
;  426          1505  4              SEARCH_DONE = 1;
;  427          1506  4              END
;  428          1507  3          ELSE
;  429          1508  4              BEGIN
;  430          1509  4          !+
;  431          1510  4          ! Check for a user frame.
;  432          1511  4          !-
;  433          1512  4
;  434          1513  5              IF (.FMP [BSF$A_HANDLER] EQLA BAS$HANDLER)
;  435          1514  4              THEN
;  436          1515  5                  BEGIN
;  437          1516  5          !+
;  438          1517  5          ! We have found the user's frame.  Get its PC.
;  439          1518  5          !-
;  440          1519  5                  USER_PC = .PREV_FMP [BSF$A_SAVED_PC];
;  441          1520  5                  SEARCH_DONE = 1;
;  442          1521  4                  END;
;  443          1522  4
;  444          1523  3              END;
;  445          1524  3
;  446          1525  2          END;
;  447          1526  2
;  448          1527  2          !+
;  449          1528  2          ! At the completion of the WHILE loop, USER_PC is set to the value
;  450          1529  2          ! to return, either 0 or the user's PC.
;  451          1530  2          !-
;  452          1531  2      RETURN (.USER_PC);
;  453          1532  1      END;                                    ! end of CALC_USER_PC


                                003C 00000 CALC_USER_PC:
                                           .WORD   Save R2,R3,R4,R5                    ; 1439
                                55  D4 00002          CLRL    SEARCH_DONE             ; 1486
                                53  D4 00004          CLRL    SEARCH_COUNTER          ; 1487
                           52   5D  D0 00006          MOVL    FP, FMP                 ; 1488
                                55  D5 00009 1$:      TSTL    SEARCH_DONE             ; 1490
                                2F  12 0000B          BNEQ    5$                      ;
                           50   52  D0 0000D          MOVL    FMP, PREV_FMP           ; 1492
                           52   0C  A0 D0 00010       MOVL    12(PREV_FMP), FMP       ; 1493
                                53  D6 00014          INCL    SEARCH_COUNTER          ; 1494
                      0000FFFF  8F  53  D1 00016      CMPL    SEARCH_COUNTER, #65535  ; 1496
                                04  14 0001D          BGTR    2$                      ;
                                52  D5 0001F          TSTL    FMP                     ;
                                04  12 00021          BNEQ    3$                      ;
                                54  D4 00023 2$:      CLRL    USER_PC                 ; 1504
```

```
                                10  11 00025          BRB     4$                                      ; 1505
                51 00000000G    00  9E 00027 3$:      MOVAB   BAS$HANDLER, R1                         ; 1513
                51              62  D1 0002E          CMPL    (FMP), R1                               ;
                                D6  12 00031          BNEQ    1$                                      ;
                54        10    A0  D0 00033          MOVL    16(PREV_FMP), USER_PC                   ; 1519
                55              01  D0 00037 4$:      MOVL    #1, SEARCH_DONE                         ; 1520
                                CD  11 0003A          BRB     1$                                      ; 1490
                50              54  D0 0003C 5$:      MOVL    USER_PC, R0                             ; 1531
                                    04 0003F          RET                                             ; 1532

; Routine Size:  64 bytes,    Routine Base:  _BAS$CODE + 041E

;  454          1533  1
```

```
;  456          1534  1 GLOBAL ROUTINE BAS$$SIGNAL_IO (                      ! Signal an I/O error
;  457          1535  1         OPEN_FLAG                                    ! Error code or translation guide
;  458          1536  1     ) : CALL_CCB NOVALUE =
;  459          1537  1
;  460          1538  1 !++
;  461          1539  1 ! FUNCTIONAL DESCRIPTION:
;  462          1540  1 !
;  463          1541  1 !       Signals a BASIC I/O error.  If requested, the error number
;  464          1542  1 !       is determined by examining the RMS error codes.
;  465          1543  1 !
;  466          1544  1 ! FORMAL PARAMETERS:
;  467          1545  1 !
;  468          1546  1 !       OPEN_FLAG.rl.v  Either a BASIC error number or a code telling
;  469          1547  1 !                       how to translate the RMS error information.
;  470          1548  1 !
;  471          1549  1 ! IMPLICIT INPUTS:
;  472          1550  1 !
;  473          1551  1 !       Various fields of the LUB and FAB.
;  474          1552  1 !
;  475          1553  1 ! IMPLICIT OUTPUTS:
;  476          1554  1 !
;  477          1555  1 !       NONE
;  478          1556  1 !
;  479          1557  1 ! ROUTINE VALUE:
;  480          1558  1 ! COMPLETION CODES:
;  481          1559  1 !
;  482          1560  1 !       NONE
;  483          1561  1 !
;  484          1562  1 ! SIDE EFFECTS:
;  485          1563  1 !
;  486          1564  1 !       May never return to its caller, depending on the severity
;  487          1565  1 !       of the error.
;  488          1566  1 !
;  489          1567  1 !--
;  490          1568  1
;  491          1569  2     BEGIN
;  492          1570  2
;  493          1571  2     EXTERNAL REGISTER
;  494          1572  2         CCB : REF BLOCK [0, BYTE];
;  495          1573  2
;  496          1574  2     LOCAL
;  497          1575  2         BASIC_ERR_CODE : BLOCK [%UPVAL, BYTE],  ! The 32-bit BASIC error code
;  498          1576  2         FILE_NAME_DESC : BLOCK [8, BYTE],       ! Descriptor for file name
;  499          1577  2         USER_PC,                                ! The user's PC, determined by scanning the stack
;  500          1578  2         CHAN,                                   ! The BASIC channel number
;  501          1579  2         STS,                                    ! RMS completion status code
;  502          1580  2         STV;                                    ! RMS status value
;  503          1581  2
;  504          1582  2 !+
;  505          1583  2 ! If this is a "memory" operation, just call BAS$$SIGNAL.
;  506          1584  2 !-
;  507          1585  2
;  508          1586  3     IF (.CCB [ISB$V_DE_ENCODE])
;  509          1587  2     THEN
;  510          1588  3         BEGIN
;  511          1589  3         BAS$$SIGNAL (.OPEN_FLAG);
;  512          1590  3         RETURN;
```

```
;  513          1591  2          END;
;  514          1592
;  515          1593  2      IF (.OPEN_FLAG GEQ 0)
;  516          1594  2      THEN
;  517          1595  2          BEGIN
;  518          1596  !+
;  519          1597  ! This is a BASIC I/O error, convert to a 32-bit VMS error code.
;  520          1598  !-
;  521          1599  2          STS = 0;
;  522          1600  2          STV = 0;
;  523          1601  2          BASIC_ERR_CODE = BAS$$COND_VAL (.OPEN_FLAG);
;  524          1602  2          END
;  525          1603  2      ELSE
;  526          1604  2          BEGIN
;  527          1605  !+
;  528          1606  ! This is an RMS error, we must compute the BASIC error number.
;  529          1607  ! Obtain the STS and STV values from the RAB or FAB.
;  530          1608  !-
;  531          1609  3
;  532          1610  4          IF (.OPEN_FLAG NEQ BAS$K_IOERR_OPE)
;  533          1611  3          THEN
;  534          1612  4              BEGIN
;  535          1613  4              STS = .CCB [RAB$L_STS];
;  536          1614  4              STV = .CCB [RAB$L_STV];
;  537          1615  3              END;
;  538          1616  3
;  539          1617  4          IF ((.OPEN_FLAG EQL BAS$K_IOERR_OPE) OR (.STS AND (.CCB [LUB$A_FAB] NEQA 0)))
;  540          1618  3          THEN
;  541          1619  4              BEGIN
;  542          1620  4
;  543          1621  4              LOCAL
;  544          1622  4                  FAB : REF BLOCK [0, BYTE];
;  545          1623  4
;  546          1624  4              FAB = .CCB [LUB$A_FAB];
;  547          1625  4              STS = .FAB [FAB$L_STS];
;  548          1626  4              STV = .FAB [FAB$L_STV];
;  549          1627  3              END;
;  550          1628  3
;  551          1629  3  !+
;  552          1630  3  ! Compute the BASIC error code corresponding to the RMS error.
;  553          1631  3  !-
;  554          1632  3          BASIC_ERR_CODE = BAS$$COND_VAL (TRANSLATE_RMS (.STS, .STV,
;  555          1633  3                  (IF (.OPEN_FLAG NEQ BAS$K_IOERR_REC) THEN 1 ELSE 0)));
;  556          1634  2          END;
;  557          1635  2
;  558          1636  2  !+
;  559          1637  2  ! Compute the BASIC channel number.
;  560          1638  2  !-
;  561          1639  2
;  562          1640  2      IF (.CCB [LUB$W_LUN] LSS 0) THEN CHAN = 0 ELSE CHAN = .CCB [LUB$W_LUN];
;  563          1641  2
;  564          1642  2  !+
;  565          1643  2  ! Compute user PC
;  566          1644  2  !-
;  567          1645  2      USER_PC = CALC_USER_PC ();
;  568          1646  2  !+
;  569          1647  2  ! Build a pointer to the file name from the LUB.
```

```
;  570          1648  2 !-
;  571          1649  2      FILE_NAME_DESC [DSC$A_POINTER] = .CCB [LUB$A_RSN];
;  572          1650  2      FILE_NAME_DESC [DSC$W_LENGTH] = .CCB [LUB$B_RSL];
;  573          1651  2      FILE_NAME_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;       ! Scalar string
;  574          1652  2      FILE_NAME_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;       ! ASCII text
;  575          1653  2 !+
;  576          1654  2 ! Signal an error.
;  577          1655  2 !-
;  578          1656  2      LIB$SIGNAL (.BASIC_ERR_CODE,                       ! The BASIC error code
;  579          1657  2          0,                                             ! No FAO arguments
;  580          1658  2          BAS$_ON_CHAFIL,                                ! " on channel n for file aaa at user PC xxx "
;  581          1659  2          3,                                             ! Three FAO arguments
;  582          1660  2          .CHAN,                                         ! BASIC Channel number of error
;  583          1661  2          FILE_NAME_DESC,                                ! File name
;  584          1662  2          .USER_PC,                                      ! User PC
;  585          1663  2          .STS,                                          ! First longword of RMS status
;  586          1664  2          .STV                                           ! Second longword of RMS status
;  587          1665  2      );
;  588          1666  2 !+
;  589          1667  2 ! All done.
;  590          1668  2 !-
;  591          1669  1      END;                                               ! end of BAS$$SIGNAL_IO


                                003C 00000          .ENTRY  BAS$$SIGNAL_IO, Save R2,R3,R4,R5        ; 1534
                     5E         08   C2 00002          SUBL2   #8, SP                                  ;
0B         96        AB         06   E1 00005          BBC     #6, -106(CCB), 1$                       ; 1586
                                04   AC DD 0000A          PUSHL   OPEN_FLAG                           ; 1589
    00000000G        00         01   FB 0000D          CALLS   #1, BAS$$SIGNAL                         ;
                                     04 00014          RET                                             ; 1588
                     52         04   AC D0 00015 1$:    MOVL    OPEN_FLAG, R2                           ; 1593
                                06   19 00019          BLSS    2$                                      ;
                                53   7C 0001B          CLRQ    STV                                     ; 1600
                                52   DD 0001D          PUSHL   R2                                      ; 1601
                                48   11 0001F          BRB     8$                                      ;
    FFFFFFFE    8F              52   D1 00021 2$:      CMPL    R2, #-2                                 ; 1610
                                08   13 00028          BEQL    3$                                      ;
                     54         08   AB D0 0002A          MOVL    8(CCB), STS                         ; 1613
                     53         0C   AB D0 0002E          MOVL    12(CCB), STV                        ; 1614
    FFFFFFFE    8F              52   D1 00032 3$:      CMPL    R2, #-2                                 ; 1617
                                08   13 00039          BEQL    4$                                      ;
                     11         54   E9 0003B          BLBC    STS, 5$                                 ;
                                E8   AB D5 0003E          TSTL    -24(CCB)                            ;
                                0C   13 00041          BEQL    5$                                      ;
                     50         E8   AB D0 00043 4$:      MOVL    -24(CCB), FAB                       ; 1624
                     54         08   A0 D0 00047          MOVL    8(FAB), STS                         ; 1625
                     53         0C   A0 D0 0004B          MOVL    12(FAB), STV                        ; 1626
    FFFFFFFF    8F              52   D1 0004F 5$:      CMPL    R2, #-1                                 ; 1633
                                04   13 00056          BEQL    6$                                      ;
                                01   DD 00058          PUSHL   #1                                      ;
                                02   11 0005A          BRB     7$                                      ;
                                7E   D4 0005C 6$:      CLRL    -(SP)                                   ;
                                53   DD 0005E 7$:      PUSHL   STV                                     ; 1632
                                54   DD 00060          PUSHL   STS                                     ;
```

```
                FB3B   CF              03  FB 00062          CALLS   #3, TRANSLATE_RMS                    ;
                                        50  DD 00067          PUSHL   R0                                  ;
          00000000G    00              01  FB 00069 8$:      CALLS   #1, BAS$$COND_VAL                   ;
                       55              50  D0 00070          MOVL    R0, BASIC_ERR_CODE                  ;
                                 C6    AB  B5 00073          TSTW    -58(CCB)                            ; 1640
                                       04  18 00076          BGEQ    9$                                  ;
                                       52  D4 00078          CLRL    CHAN                                ;
                                       04  11 0007A          BRB     10$                                 ;
                       52        C6    AB  32 0007C 9$:      CVTWL   -58(CCB), CHAN                      ;
                FF3B   CF              00  FB 00080 10$:     CALLS   #0, CALC_USER_PC                    ; 1645
                  04   AE        F8    AB  D0 00085          MOVL    -8(CCB), FILE_NAME_DESC+4           ; 1649
                       6E        F7    AB  9B 0008A          MOVZBW  -9(CCB), FILE_NAME_DESC             ; 1650
                  02   AE      010E    8F  B0 0008E          MOVW    #270, FILE_NAME_DESC+2              ; 1652
                                       53  DD 00094          PUSHL   STV                                 ; 1664
                                       11  BB 00096          PUSHR   #^M<R0,R4>                          ; 1662
                                 0C    AE  9F 00098          PUSHAB  FILE_NAME_DESC                      ; 1656
                                       52  DD 0009B          PUSHL   CHAN                                ; 1660
                                       03  DD 0009D          PUSHL   #3                                  ; 1656
                             00000000G 8F  DD 0009F          PUSHL   #BAS$_ON_CHAFIL                     ;
                                       7E  D4 000A5          CLRL    -(SP)                               ;
                                       55  DD 000A7          PUSHL   BASIC_ERR_CODE                      ;
          00000000G    00              09  FB 000A9          CALLS   #9, LIB$SIGNAL                      ;
                                       04  000B0          RET                                         ; 1669

; Routine Size:  177 bytes,    Routine Base:  _BAS$CODE + 045E

;  592          1670  1
```

```
;  594          1671  1 GLOBAL ROUTINE BAS$$STOP_IO (                    ! Signal a fatal I/O error
;  595          1672  1         OPEN_FLAG                                ! Error code or translation guide
;  596          1673  1     ) : CALL_CCB NOVALUE =
;  597          1674  1
;  598          1675  1 !++
;  599          1676  1 ! FUNCTIONAL DESCRIPTION:
;  600          1677  1 !
;  601          1678  1 !       Signals a fatal BASIC I/O error.  If requested, the error number
;  602          1679  1 !       is determined by examining the RMS error codes.
;  603          1680  1 !
;  604          1681  1 ! FORMAL PARAMETERS:
;  605          1682  1 !
;  606          1683  1 !       OPEN_FLAG.rl.v  Either a BASIC error number or a code telling
;  607          1684  1 !                       how to translate the RMS error information.
;  608          1685  1 !
;  609          1686  1 ! IMPLICIT INPUTS:
;  610          1687  1 !
;  611          1688  1 !       Various fields of the LUB and FAB.
;  612          1689  1 !
;  613          1690  1 ! IMPLICIT OUTPUTS:
;  614          1691  1 !
;  615          1692  1 !       NONE
;  616          1693  1 !
;  617          1694  1 ! ROUTINE VALUE:
;  618          1695  1 ! COMPLETION CODES:
;  619          1696  1 !
;  620          1697  1 !       NONE
;  621          1698  1 !
;  622          1699  1 ! SIDE EFFECTS:
;  623          1700  1 !
;  624          1701  1 !       Never returns to its caller.
;  625          1702  1 !
;  626          1703  1 !--
;  627          1704  1
;  628          1705  2     BEGIN
;  629          1706  2
;  630          1707  2     EXTERNAL REGISTER
;  631          1708  2         CCB : REF BLOCK [0, BYTE];
;  632          1709  2
;  633          1710  2     LOCAL
;  634          1711  2         BASIC_ERR_CODE : BLOCK [%UPVAL, BYTE],   ! The 32-bit BASIC error code
;  635          1712  2         FILE_NAME_DESC : BLOCK [8, BYTE],        ! Descriptor for file name
;  636          1713  2         USER_PC,                                 ! The user's PC, determined by scanning the stack
;  637          1714  2         CHAN,                                    ! The BASIC channel number
;  638          1715  2         STS,                                     ! RMS completion status code
;  639          1716  2         STV;                                     ! RMS status value
;  640          1717  2
;  641          1718  2 !+
;  642          1719  2 ! If this is a "memory" operation, just call BAS$$STOP.
;  643          1720  2 !-
;  644          1721  2
;  645          1722  3     IF (.CCB [ISB$V_DE_ENCODE])
;  646          1723  2     THEN
;  647          1724  3         BEGIN
;  648          1725  3         BAS$$STOP (.OPEN_FLAG);
;  649          1726  3         RETURN;
;  650          1727  2         END;
```

```
;  651          1728  2
;  652          1729  3      IF (.OPEN_FLAG GEQ 0)
;  653          1730  2      THEN
;  654          1731  3          BEGIN
;  655          1732  3  !+
;  656          1733  3  ! This is a BASIC I/O error, convert to a 32-bit VMS error code.
;  657          1734  3  !-
;  658          1735  3          STS = 0;
;  659          1736  3          STV = 0;
;  660          1737  3          BASIC_ERR_CODE = BAS$$COND_VAL (.OPEN_FLAG);
;  661          1738  3          END
;  662          1739  2      ELSE
;  663          1740  3          BEGIN
;  664          1741  3  !+
;  665          1742  3  ! This is an RMS error, we must compute the BASIC error number.
;  666          1743  3  ! Obtain the STS and STV values from the RAB or FAB.
;  667          1744  3  !-
;  668          1745  3
;  669          1746  4          IF (.OPEN_FLAG NEQ BAS$K_IOERR_OPE)
;  670          1747  3          THEN
;  671          1748  4              BEGIN
;  672          1749  4              STS = .CCB [RAB$L_STS];
;  673          1750  4              STV = .CCB [RAB$L_STV];
;  674          1751  3              END;
;  675          1752  3
;  676          1753  4          IF ((.OPEN_FLAG EQL BAS$K_IOERR_OPE) OR (.STS AND (.CCB [LUB$A_FAB] NEQA 0)))
;  677          1754  3          THEN
;  678          1755  4              BEGIN
;  679          1756  4
;  680          1757  4              LOCAL
;  681          1758  4                  FAB : REF BLOCK [0, BYTE];
;  682          1759  4
;  683          1760  4              FAB = .CCB [LUB$A_FAB];
;  684          1761  4              STS = .FAB [FAB$L_STS];
;  685          1762  4              STV = .FAB [FAB$L_STV];
;  686          1763  3              END;
;  687          1764  3
;  688          1765  3  !+
;  689          1766  3  ! Compute the BASIC error code corresponding to the RMS error.
;  690          1767  3  !-
;  691          1768  3          BASIC_ERR_CODE = BAS$$COND_VAL (TRANSLATE_RMS (.STS, .STV,
;  692          1769  3                  (IF (.OPEN_FLAG NEQ BAS$K_IOERR_REC) THEN 1 ELSE 0)));
;  693          1770  2          END;
;  694          1771  2
;  695          1772  2  !+
;  696          1773  2  ! Compute the BASIC channel number.
;  697          1774  2  !-
;  698          1775  2
;  699          1776  2      IF (.CCB [LUB$W_LUN] LSS 0) THEN CHAN = 0 ELSE CHAN = .CCB [LUB$W_LUN];
;  700          1777  2
;  701          1778  2  !+
;  702          1779  2  ! Compute user PC
;  703          1780  2  !-
;  704          1781  2      USER_PC = CALC_USER_PC ();
;  705          1782  2  !+
;  706          1783  2  ! Build a pointer to the file name from the LUB.
;  707          1784  2  !-
```

```
;  708          1785  2     FILE_NAME_DESC [DSC$A_POINTER] = .CCB [LUB$A_RSN];
;  709          1786  2     FILE_NAME_DESC [DSC$W_LENGTH] = .CCB [LUB$B_RSL];
;  710          1787  2     FILE_NAME_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;       ! Scalar string
;  711          1788  2     FILE_NAME_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;       ! ASCII text
;  712          1789  2 !+
;  713          1790  2 ! Signal a fatal error.
;  714          1791  2 !-
;  715          1792  2     LIB$STOP (.BASIC_ERR_CODE,                          ! The BASIC error code
;  716          1793  2         0,                                              ! No FAO arguments
;  717          1794  2         BAS$_ON_CHAFIL,                                 ! " on channel n for file aaa at user PC xxx "
;  718          1795  2         3,                                              ! Three FAO arguments
;  719          1796  2         .CHAN,                                          ! BASIC Channel number  f error
;  720          1797  2         FILE_NAME_DESC,                                 ! File name
;  721          1798  2         .USER_PC,                                       ! User PC
;  722          1799  2         .STS,                                           ! First longword of RMS status
;  723          1800  2         .STV                                            ! Second longword of RMS status
;  724          1801  2     );
;  725          1802  2 !+
;  726          1803  2 ! All done.
;  727          1804  2 !-
;  728          1805  1     END;                                                ! end of BAS$$STOP_IO


                                003C 00000          .ENTRY  BAS$$STOP_IO, Save R2,R3,R4,R5                  ; 1671
                           5E           08  C2 00002  SUBL2   #8, SP                                        ;
                    0B      96      AB  06  E1 00005  BBC     #6, -106(CCB), 1$                             ; 1722
                                    04  AC  DD 0000A  PUSHL   OPEN_FLAG                                     ; 1725
                00000000G   00          01  FB 0000D  CALLS   #1, BAS$$STOP                                ;
                                        04  00014      RET                                                 ; 1724
                            52      04  AC  D0 00015  1$:     MOVL    OPEN_FLAG, R2                       ; 1729
                                        06  19 00019      BLSS    2$                                      ;
                                        53  7C 0001B      CLRQ    STV                                     ; 1736
                                        52  DD 0001D      PUSHL   R2                                      ; 1737
                                        48  11 0001F      BRB     8$                                      ;
                FFFFFFFE    8F          52  D1 00021  2$:     CMPL    R2, #-2                             ; 1746
                                        08  13 00028      BEQL    3$                                      ;
                            54      08  AB  D0 0002A      MOVL    8(CCB), STS                             ; 1749
                            53      0C  AB  D0 0002E      MOVL    12(CCB), STV                            ; 1750
                FFFFFFFE    8F          52  D1 00032  3$:     CMPL    R2, #-2                             ; 1753
                                        08  13 00039      BEQL    4$                                      ;
                            11          54  E9 0003B      BLBC    STS, 5$                                 ;
                                    E8  AB  D5 0003E      TSTL    -24(CCB)                                ;
                                        0C  13 00041      BEQL    5$                                      ;
                            50      E8  AB  D0 00043  4$:     MOVL    -24(CCB), FAB                       ; 1760
                            54      08  A0  D0 00047      MOVL    8(FAB), STS                             ; 1761
                            53      0C  A0  D0 0004B      MOVL    12(FAB), STV                            ; 1762
                FFFFFFFF    8F          52  D1 0004F  5$:     CMPL    R2, #-1                             ; 1769
                                        04  13 00056      BEQL    6$                                      ;
                                        01  DD 00058      PUSHL   #1                                      ;
                                        02  11 0005A      BRB     7$                                      ;
                                        7E  D4 0005C  6$:     CLRL    -(SP)                               ;
                                        53  DD 0005E  7$:     PUSHL   STV                                 ; 1768
                                        54  DD 00060      PUSHL   STS                                     ;
                        FA8A    CF          03  FB 00062      CALLS   #3, TRANSLATE_RMS                   ;
```

```
                                        50  DD 00067          PUSHL   R0                                      ;
                00000000G   00          01  FB 00069 8$:      CALLS   #1, BAS$$COND_VAL                       ;
                            55          50  D0 00070          MOVL    R0, BASIC_ERR_CODE                      ;
                                  C6    AB  B5 00073          TSTW    -58(CCB)                                ; 1776
                                        04  18 00076          BGEQ    9$                                      ;
                                        52  D4 00078          CLRL    CHAN                                    ;
                                        04  11 0007A          BRB     10$                                     ;
                            52        C6    AB  32 0007C 9$:      CVTWL   -58(CCB), CHAN                      ;
                          FE8A        CF    00  FB 00080 10$:     CALLS   #0, CALC_USER_PC                    ; 1781
                            04        AE    F8  AB  D0 00085      MOVL    -8(CCB), FILE_NAME_DESC+4           ; 1785
                            6E        F7    AB  9B 0008A          MOVZBW  -9(CCB), FILE_NAME_DESC             ; 1786
                            02        AE  010E  8F  B0 0008E      MOVW    #270, FILE_NAME_DESC+2              ; 1788
                                        53  DD 00094          PUSHL   STV                                     ; 1800
                                        11  BB 00096          PUSHR   #^M<R0,R4>                              ; 1798
                                        0C  AE  9F 00098      PUSHAB  FILE_NAME_DESC                          ; 1792
                                        52  DD 0009B          PUSHL   CHAN                                    ; 1796
                                        03  DD 0009D          PUSHL   #3                                      ; 1792
                                00000000G   8F  DD 0009F      PUSHL   #BAS$_ON_CHAFIL                         ;
                                        7E  D4 000A5          CLRL    -(SP)                                   ;
                                        55  DD 000A7          PUSHL   BASIC_ERR_CODE                          ;
                00000000G   00          09  FB 000A9          CALLS   #9, LIB$STOP                            ;
                                        04  000B0          RET                                             ; 1805

; Routine Size:  177 bytes,    Routine Base:  _BAS$CODE + 050F

;  729          1806  1
```

```
;  731          1807  1 GLOBAL ROUTINE BAS$$STOP_RMS (                    ! Signal a fatal I/O error
;  732          1808  1         FILE_NAME,                                 ! The file on which the error happened
;  733          1809  1         STS,                                       ! The RMS STS value
;  734          1810  1         STV                                        ! The RMS STV value
;  735          1811  1     ) : NOVALUE =
;  736          1812  1
;  737          1813  1 !++
;  738          1814  1 ! FUNCTIONAL DESCRIPTION:
;  739          1815  1 !
;  740          1816  1 !       Signals a fatal BASIC I/O error.  The error number
;  741          1817  1 !       is determined by examining the RMS error codes.
;  742          1818  1 !
;  743          1819  1 ! FORMAL PARAMETERS:
;  744          1820  1 !
;  745          1821  1 !       FILE_NAME.rt.dx The name of the file which got the error
;  746          1822  1 !       STS.rl.v        The RMS STS value
;  747          1823  1 !       STV.rl.v        The RMS STV value
;  748          1824  1 !
;  749          1825  1 ! IMPLICIT INPUTS:
;  750          1826  1 !
;  751          1827  1 !       NONE
;  752          1828  1 !
;  753          1829  1 ! IMPLICIT OUTPUTS:
;  754          1830  1 !
;  755          1831  1 !       NONE
;  756          1832  1 !
;  757          1833  1 ! ROUTINE VALUE:
;  758          1834  1 ! COMPLETION CODES:
;  759          1835  1 !
;  760          1836  1 !       NONE
;  761          1837  1 !
;  762          1838  1 ! SIDE EFFECTS:
;  763          1839  1 !
;  764          1840  1 !       Never returns to its caller.
;  765          1841  1 !
;  766          1842  1 !--
;  767          1843  1
;  768          1844  2     BEGIN
;  769          1845  2
;  770          1846  2     LOCAL
;  771          1847  2         BASIC_ERR_CODE : BLOCK [%UPVAL, BYTE];  ! The 32-bit BASIC error code
;  772          1848  2
;  773          1849  2 !+
;  774          1850  2 ! Compute the BASIC error code corresponding to the RMS error.
;  775          1851  2 !-
;  776          1852  2     BASIC_ERR_CODE = BAS$$COND_VAL (TRANSLATE_RMS (.STS, .STV, 0));
;  777          1853  2 !+
;  778          1854  2 ! Signal a fatal error.
;  779          1855  2 !-
;  780          1856  2     LIB$STOP (.BASIC_ERR_CODE,                      ! The BASIC error code
;  781          1857  2         0,                                          ! No FAO arguments
;  782          1858  2         BAS$_FORFILUSE,                             ! "for file"...
;  783          1859  2         2,                                          ! Two FAO arguments
;  784          1860  2         .FILE_NAME,                                 ! File name
;  785          1861  2         CALC_USER_PC (),                            ! User PC
;  786          1862  2         .STS,                                       ! First longword of RMS status
;  787          1863  2         .STV                                        ! Second longword of RMS status
```

```
;  788          1864  2     );
;  789          1865  2 !+
;  790          1866  2 ! All done.
;  791          1867  2 !-
;  792          1868  1     END;                                    ! end of BAS$$STOP_RMS



                                        0004 00000         .ENTRY  BAS$$STOP_RMS, Save R2                          ; 1807
                                     7E D4 00002         CLRL    -(SP)                                           ; 1852
                                  7E    08 AC  7D 00004  MOVQ    STS, -(SP)                                      ;
                                FA33    CF 03  FB 00008  CALLS   #3, TRANSLATE_RMS                               ;
                                        50 DD 0000D      PUSHL   R0                                              ;
                           00000000G    00 01  FB 0000F  CALLS   #1, BAS$$COND_VAL                               ;
                                        52 50  D0 00016  MOVL    R0, BASIC_ERR_CODE                              ;
                                  7E    08 AC  7D 00019  MOVQ    STS, -(SP)                                      ; 1862
                                FE3C    CF 00  FB 0001D  CALLS   #0, CALC_USER_PC                                ; 1861
                                        50 DD 00022      PUSHL   R0                                              ;
                                     04 AC  DD 00024     PUSHL   FILE_NAME                                       ; 1860
                                        0? DD 00027      PUSHL   #2                                              ; 1856
                                00000000G  8F DD 00029   PUSHL   #BAS$_FORFILUSE                                 ;
                                     7E D4 0002F         CLRL    -(SP)                                           ;
                                     52 DD 00031         PUSHL   BASIC_ERR_CODE                                  ;
                           00000000G    00 08  FB 00033  CALLS   #8, LIB$STOP                                    ;
                                        04 0003A         RET                                                     ; 1868

; Routine Size:  59 bytes,    Routine Base:  _BAS$CODE + 05C0

;  793          1869  1
;  794          1870  1 END                                         ! end of module BAS$$SIGNAL_IO
;  795          1871  1
;  796          1872  0 ELUDOM
```

PSECT SUMMARY

Name	Bytes	Attributes
_BAS$CODE	1531	NOVEC,NOWRT, RD , EXE, SHR, LCL, REL, CON, PIC,ALIGN(2)

Library Statistics

File	Symbols Total	Symbols Loaded	Symbols Percent	Pages Mapped	Processing Time
_255DUA28:[SYSLIB]STARLET.L32;1	9776	85	0	581	00:01.2

```
;                                       COMMAND QUALIFIERS

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

; Size:         1531 code + 0 data bytes
; Run Time:       00:30.2
; Elapsed Time:   01:06.4
; Lines/CPU Min:   3714
; Lexemes/CPU-Min: 19748
; Memory Used:  259 pages
; Compilation Complete
```