```
FFFFFFFFFFFFFFF     000000000      RRRRRRRRRRRR     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
FFFFFFFFFFFFFFF     000000000      RRRRRRRRRRRR     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
FFFFFFFFFFFFFFF     000000000      RRRRRRRRRRRR     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
FFF              000         000   RRR         RRR  RRR         RRR        TTT        LLL
FFF              000         000   RRR         RRR  RRR         RRR        TTT        LLL
FFF              000         000   RRR         RRR  RRR         RRR        TTT        LLL
FFF              000         000   RRR         RRR  RRR         RRR        TTT        LLL
FFF              000         000   RRR         RRR  RRR         RRR        TTT        LLL
FFF              000         000   RRR         RRR  RRR         RRR        TTT        LLL
FFFFFFFFFFFF     000         000   RRRRRRRRRRRR     RRRRRRRRRRRR           TTT        LLL
FFFFFFFFFFFF     000         000   RRRRRRRRRRRR     RRRRRRRRRRRR           TTT        LLL
FFFFFFFFFFFF     000         000   RRRRRRRRRRRR     RRRRRRRRRRRR           TTT        LLL
FFF              000         000   RRR   RRR        RRR   RRR              TTT        LLL
FFF              000         000   RRR   RRR        RRR   RRR              TTT        LLL
FFF              000         000   RRR   RRR        RRR   RRR              TTT        LLL
FFF              000         000   RRR      RRR     RRR      RRR           TTT        LLL
FFF              000         000   RRR      RRR     RRR      RRR           TTT        LLL
FFF              000         000   RRR      RRR     RRR      RRR           TTT        LLL
FFF                 000000000      RRR         RRR  RRR         RRR        TTT        LLLLLLLLLLLLLLL
FFF                 000000000      RRR         RRR  RRR         RRR        TTT        LLLLLLLLLLLLLLL
FFF                 000000000      RRR         RRR  RRR         RRR        TTT        LLLLLLLLLLLLLLL
```

```
                                                          D 16
**FILE**ID**FORSIGNAL
```

```
FFFFFFFFFF    OOOOOO    RRRRRRRR      SSSSSSSS    IIIIII      GGGGGGGG  NN      NN    AAAAAA    LL
FFFFFFFFFF    OOOOOO    RRRRRRRR      SSSSSSSS    IIIIII      GGGGGGGG  NN      NN    AAAAAA    LL
FF          OO      OO  RR      RR  SS              II      GG          NN      NN  AA      AA  LL
FF          OO      OO  RR      RR  SS              II      GG          NN      NN  AA      AA  LL
FF          OO      OO  RR      RR  SS              II      GG          NNNN    NN  AA      AA  LL
FF          OO      OO  RR      RR  SS              II      GG          NNNN    NN  AA      AA  LL
FFFFFFFF    OO      OO  RRRRRRRR      SSSSSS        II      GG          NN  NN  NN  AA      AA  LL
FFFFFFFF    OO      OO  RRRRRRRR      SSSSSS        II      GG          NN  NN  NN  AA      AA  LL
FF          OO      OO  RR  RR              SS      II      GG  GGGGGG  NN    NNNN  AAAAAAAAAA  LL
FF          OO      OO  RR  RR              SS      II      GG  GGGGGG  NN    NNNN  AAAAAAAAAA  LL
FF          OO      OO  RR    RR            SS      II      GG      GG  NN      NN  AA      AA  LL              ....
FF          OO      OO  RR    RR            SS      II      GG      GG  NN      NN  AA      AA  LL              ....
FF            OOOOOO    RR      RR  SSSSSSSS      IIIIII      GGGGGG    NN      NN  AA      AA  LLLLLLLLLL      ....
FF            OOOOOO    RR      RR  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
```

```
0 MODULE FOR$$SIGNAL (%TITLE'FORTRAN SIGNAL, SIGNAL_STOP and SIG_NO_LUB'
0         IDENT = '1-007' ! File: FORSIGNAL.B32   Edit: SBL1007
0                 ) =
1 BEGIN
1
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1 !
1
1 !++
1 ! FACILITY:  FORTRAN Support Library
1 !
1 ! ABSTRACT:
1 !
1 !       FORTRAN support routines to convert FORTRAN error code
1 !       to 32-bit VAX error code, and SIGNAL or SIGNAL_STOP
1 !       extra information in format compatible for SYS$PUT_MESSAGE:
1 !
1 ! ENVIRONMENT:  User Mode - AST re-entrant
1 !               Note: this module is both shared and non-shared.
1 !               If compatibility routine calls it, a non-shared copy is included.
1 !               Hence, JSB to FOR$$CB_GET instead of accessing OTS$$A_CUR_LUB directly.
1 !
1 ! AUTHOR:  Thomas N. Hastings, CREATION DATE:  8-Aug-1977
1 !
1 ! MODIFIED BY:
1 !
1 !       Thomas N. Hastings, 8-Aug-1977: VERSION 0
1 !       Steven B. Lionel, VAX/VMS V2.0
1 ! [Previous edit history removed.  SBL 10-Nov-1980]
1 ! 1-001 - Update version number and copyright notice.  JBS 16-NOV-78
1 ! 1-002 - Change LUB$B_LUN to LUB$W_LUN.  JBS 05-DEC-78
1 ! 1-003 - Change REQUIRE file names from FOR... to OTS...  JBS 06-DEC-78
1 ! 1-004 - Get filename from FAB if all else fails.  SBL 29-Aug-1979
1 ! 1-005 - Add optional FAB argument to FOR$$SIG_NO_LUB.  SBL 7-OCT-1979
1 ! 1-006 - Allow extra FAO arguments and conditions to be passed to
```

```
;    58          0058  1 !             FOR$$SIGNAL and FOR$$SIGNAL_STO.  Remove debugging macros, no longer
;    59          0059  1 !             used.  SBL 10-Nov-1980
;    60          0060  1 ! *** First post-V3.0 edit ***
;    61          0061  1 ! 1-007 - Use prologue file.  SBL 20-Jan-1983
;    62          0062  1 !--
```

```
!
! PROLOGUE FILE:
!

REQUIRE 'RTLIN:FORPROLOG';                          ! FOR$ Definitions

!
! TABLE OF CONTENTS:
!

FORWARD ROUTINE
        FOR$$SIGNAL: NOVALUE,                       ! SIGNAL 32-bit error code and LUB data
        FOR$$SIGNAL_STO: NOVALUE,                   ! SIGNAL_STOP 32-bit error code and LUB data
        FOR$$SIG_FATINT: NOVALUE,                   ! SIGNAL_STOP OTS$_FATINTERR (FATAL
                                                    ! INTERNAL ERROR IN RUN-TIME LIBRAY)
        FOR$$SIG_DATCOR: NOVALUE,                   ! SIGNAL_STOP OTS$_INTDATCOR (INTERNAL
                                                    ! DATA CORRUPTED IN RUN-TIME LIBRARY)
        DO_SIGNAL: NOVALUE,                         ! Do the work for FOR$$SIGNAL, FOR$$SIGNAL_STO
        FOR$$SIG_NO_LUB: NOVALUE,                   !  SIGNAL_STOP with no LUB setup.
        COND_VALUE;                                 ! Return 32-bit condition value given FORTRAN error #

!
! MACROS:
!

!
! EQUATED SYMBOLS:
!
LITERAL
        K_NO_FAO_SIGARG = 3;        ! No. of FAO args in signal arg list
                                    ! used by SYS$PUT_MESSAGE
!
! OWN STORAGE:
!
!       NONE

!
! EXTERNAL REFERENCES:
!

!+
! MAINTENANCE NOTE:  Since this module is called by FORTRAN compatibility
! routines which are un-shared and the entry points are not vectored,
! a separate copy of this module is linked with the user program when
! the user calls a FORTRAN compatibility routine.  In order to prevent
! data truncation errors from the linker, all external references are
! of addressing mode general (rather than word displacement) even for
! the same PSECT.
!-


EXTERNAL ROUTINE
        FOR$$CB_GET: JSB_CB_GET NOVALUE ADDRESSING_MODE (GENERAL),      ! Set CCB to adr. of current LUB/ISB
        FOR$$ERRSNS_SAV: NOVALUE ADDRESSING_MODE (GENERAL),     ! Save error info for FOR$ERRSNS.
        LIB$SIGNAL: NOVALUE ADDRESSING_MODE(GENERAL),   ! SIGNAL error and continue
        LIB$STOP: NOVALUE ADDRESSING_MODE(GENERAL);     ! SIGNAL error and STOP
```

```
;  121          0185   1 EXTERNAL LITERAL
;  122          0186   1          OTS$_FATINTERR: UNSIGNED (%BPVAL),      ! Condition value FATAL INTERNAL ERROR IN RUN-TIME LIBRARY
;  123          0187   1          OTS$_INTDATCOR: UNSIGNED (%BPVAL);      ! Condition value INTERNAL DATA CORRUPTED IN RUN-TIME LIBRAR
;  124          0188   1
```

```
1 GLOBAL ROUTINE FOR$$SIGNAL          ! SIGNAL FORTRAN error and continue
1         :NOVALUE =                  ! No value returned.
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Signals a FORTRAN-specific error whose 32-bit condition code or
1 !       small-integer error number is the first argument.  If other arguments
1 !       are present, they represent extra FAO arguments for the first
1 !       condition and/or secondary conditions to be signalled.  See DO_SIGNAL
1 !       for more information
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       CALL FOR$$SIGNAL (fort_err_no.rc.v [,fao_args_0.rz.v, ...]
1 !                         [,secondary_msg.rc.v [,sec_fao_cnt.rl.v[, sec_fao_args.rz.v,...]])
1
1 ! FORMAL PARAMETERS:
1 !
1 !       fort_err_no     - A 32-bit FOR$ code or the small integer which is the
1 !                         error number part of a FOR$ code.
1 !       fao_args_0      - FAO arguments for this message.  The three FAO arguments
1 !                         unit number, filename and user PC are always used; if
1 !                         fao_args_0 are specified, they come before the default
1 !                         arguments.
1 !       secondary_msg   - Secondary message to be signalled.  MUST be a 32-bit code.
1 !       sec_fao_cnt     - FAO count for secondary message
1 !       sec_fao_args    - FAO arguments for secondary message
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       See DO_SIGNAL
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       See DO_SIGNAL
1 !
1 ! COMPLETION CODES:
1 !
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !
1 !       Converts FORTRAN error code to 32-bit VAX-11 error code and SIGNALs.
1 !       Saves error info in FOR$ERRSNS OWN storage.
1 !--
1
2     BEGIN
2
2     BUILTIN
2         AP;
2
2     DO_SIGNAL (.AP, LIB$SIGNAL);
2
2     RETURN
1     END;
```

```
                                                    .TITLE  FOR$$SIGNAL FORTRAN SIGNAL, SIGNAL_STOP and SIG
;                                                                       _NO_LUB
                                                    .IDENT  \1-007\

                                                    .EXTRN  FOR$$CB_GET, FOR$$ERRSNS_SAV
                                                    .EXTRN  LIB$SIGNAL, LIB$STOP
                                                    .EXTRN  OTS$_FATINTERR, OTS$_INTDATCOR

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

                                   0000 00000       .ENTRY  FOR$$SIGNAL, Save nothing                 ; 0189
                       00000000G  00  9F 00002      PUSHAB  LIB$SIGNAL                                ; 0240
                                  5C  DD 00008      PUSHL   AP                                        ;
             0000V  CF            02  FB 0000A      CALLS   #2, DO_SIGNAL                             ;
                                      04 0000F      RET                                               ; 0243

; Routine Size:  16 bytes,     Routine Base:  _FOR$CODE + 0000
;  182          0244  1 GLOBAL ROUTINE FOR$$SIGNAL_STO  ! SIGNAL_STOP FORTRAN error and STOP
;  183          0245  1         :NOVALUE =              ! No value returned.
;  184          0246  1 !++
;  185          0247  1 ! FUNCTIONAL DESCRIPTION:
;  186          0248  1 !
;  187          0249  1 !       Convert FORTRAN error code number to 32-bit VAX-11 error code.
;  188          0250  1 !       See description for FOR$$SIGNAL above which is identical,
;  189          0251  1 !       except that FOR$$SIGNAL_STO calls LIB$STOP instead of LIB$SIGNAL.
;  190          0252  1 !--
;  191          0253  1
;  192          0254  2     BEGIN
;  193          0255  2
;  194          0256  2     BUILTIN
;  195          0257  2         AP;
;  196          0258  2
;  197          0259  2     DO_SIGNAL (.AP, LIB$STOP);
;  198          0260  2
;  199          0261  2     RETURN
;  200          0262  1     END;



                                      0000 00000          .ENTRY  FOR$$SIGNAL_STO, Save nothing                        ; 0244
                      00000000G  00   9F 00002          PUSHAB  LIB$STOP                                             ; 0259
                                 5C   DD 00008          PUSHL   AP                                                   ;
               0000V  CF         02   FB 0000A          CALLS   #2, DO_SIGNAL                                        ;
                                      04 0000F          RET                                                          ; 0262

; Routine Size:  16 bytes,    Routine Base:  _FOR$CODE + 0010
;  202          0263  1 GLOBAL ROUTINE FOR$$SIG_FATINT           ! SIGNAL_STOP OTS$_FATINTERR and STOP
;  203          0264  1         :NOVALUE =               ! No value returned.
;  204          0265  1 !++
;  205          0266  1 ! FUNCTIONAL DESCRIPTION:
;  206          0267  1 !
;  207          0268  1 !       SIGNAL_STOP OTS$_FATINTERR = FATAL INTERNAL ERROR IN RUN-TIME LIBRARY.
;  208          0269  1 !       Note: the current LUB (if any) is ignored and no UNIT is printed.
;  209          0270  1 !--
;  210          0271  1
;  211          0272  2     BEGIN
;  212          0273  2     FOR$$SIG_NO_LUB (OTS$_FATINTERR);
;  213          0274  2     RETURN
;  214          0275  1     END;



                                          0000 00000          .ENTRY  FOR$$SIG_FATINT, Save nothing                         ; 0263
                           00000000G  8F    DD 00002          PUSHL   #OTS$_FATINTERR                                       ; 0273
                  0000V  CF           01    FB 00008          CALLS   #1, FOR$$SIG_NO_LUB                                   ;
                                            04 0000D          RET                                                           ; 0275

; Routine Size:  14 bytes,    Routine Base:  _FOR$CODE + 0020
;  216          0276   1 GLOBAL ROUTINE FOR$$SIG_DATCOR               ! SIGNAL_STOP OTS$_INTDATCOR and STOP
;  217          0277   1        :NOVALUE =                     ! No value returned.
;  218          0278   1 !++
;  219          0279   1 ! FUNCTIONAL DESCRIPTION:
;  220          0280   1 !
;  221          0281   1 !        SIGNAL_STOP OTS$_INTDATCOR = INTERNAL DATA CORRUPTED IN RUN-TIME LIBRARY.
;  222          0282   1 !        Note: the current LUB (if any) is ignored and no UNIT is printed.
;  223          0283   1 !--
;  224          0284   1
;  225          0285   2     BEGIN
;  226          0286   2     FOR$$SIG_NO_LUB (OTS$_INTDATCOR);
;  227          0287   2     RETURN
;  228          0288   1     END;



                                        0000 00000          .ENTRY  FOR$$SIG_DATCOR, Save nothing                        ; 0276
                          00000000G  8F   DD 00002          PUSHL   #OTS$_INTDATCOR                                      ; 0286
                 0000V  CF           01   FB 00008          CALLS   #1, FOR$$SIG_NO_LUB                                  ;
                                          04 0000D          RET                                                          ; 0288

; Routine Size:  14 bytes,    Routine Base:  _FOR$CODE + 002E
```

```
1 ROUTINE DO_SIGNAL (                   ! Internal routine to do work for FOR$$SIGNAL and FOR$$SIGNAL_STO
1           SIGNAL_LIST_ARG,            ! list of arguments to signal routine
1           SIGNAL_ROUTINE)             ! adr. of LIB$SIGNAL or LIB$STOP
1           : NOVALUE =                 ! No value returned
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Converts error code number to 32-bit VAX-11 error code.
1 !       See description of FOR$$SIGNAL above.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       SIGNAL_LIST_ARG         Contents of AP at time of call to FOR$$SIGNAL or
1 !                               FOR$$SIGNAL_STO
1 !       SIGNAL_ROUTINE          Adr. of LIB$SIGNAL or LIB$STOP
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       OTS$$A_CUR_LUB          Adr. of current LUB/ISB/RAB
1 !                               Obtained by JSB to FOR$$CB_GET.
1 !       {FAB,RAB}$L_STS         RMS error status
1 !       {FAB,RAB}$L_STV         RMS error value or operating system error code
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       {FAB,RAB}$L_STS         RMS error status - set to 0
1 !       {FAB,RAB}$L_STV         RMS error value or operating system error code - set to 0
1 !       FORTRAN error #, RMS STS, RMS STV, logical unit number saved in
1 !                               OWN storage in FOR$ERRSNS module for later
1 !                               call by user to ERRSNS.
1 !
1 ! COMPLETION CODES:
1 !
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !
1 !       Converts FORTRAN error code to 32-bit VAX-11 error code and SIGNALs.
1 !       Saves error info in FOR$ERRSNS OWN storage.
1 !--
1
```

```
  2     BEGIN
  2
  2     GLOBAL REGISTER
  2         CCB = K_CCB_REG: REF $FOR$CCB_DECL;
  2
  2     LOCAL
  2         FILE_NAME_DSC: DSC$DESCRIPTOR,  ! File name descriptor for resultant file name
  2         RABORFAB: REF BLOCK[, BYTE],
  2         STS,                            ! RMS RAB or FAB error status
  2         STV,                            ! RMS RAB or FAB error status
  2         GETMSG_VALS: VECTOR [4, BYTE],  ! Returned values from $GETMSG
  2         SIGNAL_LIST: VECTOR [20, LONG], ! Argument list to LIB$SIGNAL/LIB$STOP
  2         LIST_PTR: REF VECTOR [, LONG],  ! Pointer into signal list
  2         ARGS_PTR: REF VECTOR [, LONG],  ! pointer into SIGNAL_LIST_ARG
  2         ARG_LIST_END,                   ! Address of argument list end
  2         COND_VAL: BLOCK [4,BYTE];       ! 32-bit VAX-11 error code
  2
  2     MAP
  2         SIGNAL_LIST_ARG: REF VECTOR [, LONG];
  2
  2     BUILTIN
  2         CALLG;
  2
  2     FOR$$CB_GET ();                     ! Set CCB to adr. of current LUB/ISB/RAB
  2
  2     !+
  2     ! Convert FORTRAN error code to 32-bit VAX-11 error code.
  2     ! Conversion is done by copying FORTRAN error number to code field,
  2     ! setting the severity code to SEVERE,
  2     ! for all errors except FOR$_OUTCONERR (63='OUTPUT CONVERSION ERROR')
  2     ! which is set to ERROR instead so that image will continue
  2     ! by default since output field is flagged with ***s.
  2     ! All other continuable errors are signaled SEVERE so that user
  2     ! must take overt action in order to continue past the error.
  2     ! setting the facility code to FOR$K_FAC_NO,
  2     ! and setting the facility specific bit (STS$V_FAC_SP).
  2     !-
  2
  2     COND_VAL = COND_VALUE (.SIGNAL_LIST_ARG [1]);
  2
  2     !+
  2     ! Call $GETMSG to see how many FAO parameters it takes.
  2     !-
  2
  3         BEGIN
  3         LOCAL
  3             DSC: VECTOR [2, LONG],
  3             LEN;
  3         DSC [0] = 0;    ! Null string descriptor
  3         DSC [1] = LEN;
P 3         $GETMSG (
P 3             MSGID = .COND_VAL,
P 3             MSGLEN = LEN,
P 3             BUFADR = DSC,
P 3             FLAGS = 0,
  3             OUTADR = GETMSG_VALS);
  2         END;
```

```

        !+
        ! Compute total number of signal arguments.
        !-

        SIGNAL_LIST [0] = (.SIGNAL_LIST_ARG [0])<0,8,0> + 6;
        ARG_LIST_END = SIGNAL_LIST_ARG [0] + ((.SIGNAL_LIST_ARG [0])<0,8,0> * %UPVAL);

        !+
        ! Fill in primary condition message.
        !-

        SIGNAL_LIST [1] = .COND_VAL;
        SIGNAL_LIST [2] = .GETMSG_VALS [1]; ! Number of FAO parameters

        LIST_PTR = SIGNAL_LIST [3];
        ARGS_PTR = SIGNAL_LIST_ARG [2];

        !+
        ! Copy extra FAO arguments, if any.
        !-

        INCR I FROM 4 TO .SIGNAL_LIST [2] DO
            COPY_LONG_A (ARGS_PTR, LIST_PTR);

        !+
        ! Get RMS error status from RAB or if not error there from FAB (if any).
        ! Then set error status longwords to 0 so will not be found again.
        ! Note: this code depends on the fact that FAB$L_STS/STV have the same offsets
        ! as RAB$L_STS/STV.
        !-

        STS = 0;        ! Set initial values
        STV = 0;
        IF .CCB [LUB$W_LUN] NEQU LUB$K_LUN_ENCD      ! Not ENCODE/DECODE/internal?
        THEN
            BEGIN
            RABORFAB = .CCB;
            IF (.CCB[RAB$L_STS] OR .CCB[RAB$L_STS] EQL 0)
            THEN
                RABORFAB = .CCB[RAB$L_FAB];

            IF NOT .RABORFAB[RAB$L_STS]
            THEN
                BEGIN
                STS = .RABORFAB[RAB$L_STS];
                STV = .RABORFAB[RAB$L_STV];
                END;

            RABORFAB[RAB$L_STS] = 0;
            RABORFAB[RAB$L_STV] = 0;
            END;

        !+
        ! Save FORTRAN error number, RMS STS, RMS STV, logical unit number and VAX-11 condition value
        !-

```

```
;    386          0444  2        FOR$$ERRSNS_SAV (.COND_VAL [STS$V_CODE], .STS, .STV, .CCB[LUB$W_LUN], .COND_VAL);
;    387          0445  2
;    388          0446  2        !+
;    389          0447  2        ! Set up resultant file name descriptor that gets put in signal arg list.
;    390          0448  2        ! Note that this points at the FAB's FNM until the file is opened.
;    391          0449  2        !-
;    392          0450  2
;    393          0451  2        FILE_NAME_DSC[DSC$W_LENGTH] = .CCB[LUB$B_RSL];
;    394          0452  2        FILE_NAME_DSC[DSC$B_DTYPE] = FILE_NAME_DSC[DSC$B_CLASS] = 0;
;    395          0453  2        FILE_NAME_DSC[DSC$A_POINTER] =
;    396          0454  3            ( IF .CCB[LUB$B_RSL] EQLU 0
;    397          0455  3            THEN
;    398          0456  3                0
;    399          0457  3            ELSE
;    400          0458  2                .CCB[LUB$A_RSN]);
;    401          0459  2
;    402          0460  2        !+
;    403          0461  2        ! Insert the three default FAO arguments plus the STS and STV.
;    404          0462  2        !-
;    405          0463  2
;    406          0464  2        LIST_PTR [0] = .CCB [LUB$W_LUN];
;    407          0465  2        LIST_PTR [1] = FILE_NAME_DSC;
;    408          0466  2        LIST_PTR [2] = 0;                       ! For user PC
;    409          0467  2        LIST_PTR [3] = .STS;
;    410          0468  2        LIST_PTR [4] = .STV;
;    411          0469  2        LIST_PTR = LIST_PTR [5];
;    412          0470  2
;    413          0471  2        WHILE .ARGS_PTR LEQ .ARG_LIST_END DO
;    414          0472  2            COPY_LONG_A (ARGS_PTR, LIST_PTR);
;    415          0473  2
;    416          0474  2        !+
;    417          0475  2        ! Call LIB$STOP to STOP the error or LIB$SIGNAL to SIGNAL the error.
;    418          0476  2        !-
;    419          0477  2
;    420          0478  2        CALLG (SIGNAL_LIST, .SIGNAL_ROUTINE);
;    421          0479  2
;    422          0480  2        !+
;    423          0481  2        ! Return
;    424          0482  2        !-
;    425          0483  2
;    426          0484  2        RETURN
;    427          0485  1        END;                            ! End of FOR$$SIGNAL_STO routine


                                                                       .EXTRN  SYS$GETMSG

                                          08FC 00000 DO_SIGNAL:
                                                                       .WORD   Save R2,R3,R4,R5,R6,R7,R11                     ; 0289
                    5E          98   AE  9E 00002                      MOVAB   -104(SP), SP
                       00000000G  00  16 00006                         JSB     FOR$$CB_GET                                    ; 0353
                    52          04   AC  D0 0000C                      MOVL    SIGNAL_LIST_ARG, R2                            ; 0368
                                04   A2  DD 00010                      PUSHL   4(R2)                                          ;
             0000V  CF               01  FB 00013                      CALLS   #1, COND_VALUE                                 ;
                    54               50  D0 00018                      MOVL    R0, COND_VAL                                   ;
                                08   AE  D4 0001B                      CLRL    DSC                                            ; 0378
                0C  AE               6E  9E 0001E                      MOVAB   LEN, DSC+4                                     ; 0379
```

```
                                               04    AE  9F 00022        PUSHAB  GETMSG_VALS                          ; 0385
                                                     7E  D4 00025        CLRL    -(SP)                                ;
                                               10    AE  9F 00027        PUSHAB  DSC                                  ;
                                               0C    AE  9F 0002A        PUSHAB  LEN                                  ;
                                                     54  DD 0002D        PUSHL   COND_VAL                             ;
                          00000000G  00              05  FB 0002F        CALLS   #5, SYS$GETMSG                       ;
                                     50              62  D0 00036        MOVL    (R2), R0                             ; 0392
                                 10  AE        06    A0  9E 00039        MOVAB   6(R0), SIGNAL_LIST                   ;
                                     57            8240  DE 0003E        MOVAL   (R2)+[R0], ARG_LIST_END              ; 0393
                                 14  AE              54  D0 00042        MOVL    COND_VAL, SIGNAL_LIST+4              ; 0399
                                 18  AE        05    AE  9A 00046        MOVZBL  GETMSG_VALS+1, SIGNAL_LIST+8         ; 0400
                                     53        1C    AE  9E 0004B        MOVAB   SIGNAL_LIST+12, LIST_PTR             ; 0402
                                     52              04  C0 0004F        ADDL2   #4, ARGS_PTR                         ; 0403
                                     50              03  D0 00052        MOVL    #3, I                                ; 0409
                                                     03  11 00055        BRB     2$                                   ;
                                     83              82  D0 00057 1$:    MOVL    (ARGS_PTR)+, (LIST_PTR)+             ; 0410
                     F8              50        18    AE  F3 0005A 2$:    AOBLEQ  SIGNAL_LIST+8, I, 1$                 ;
                                                     55  7C 0005F        CLRQ    STV                                  ; 0420
                               FFFB  8F        C6    AB  B1 00061        CMPW    -58(CCB), #-5                        ; 0421
                                                     1F  13 00067        BEQL    6$                                   ;
                                     50              5B  D0 00069        MOVL    CCB, RABORFAB                        ; 0424
                                     05        08    AB  E8 0006C        BLBS    8(CCB), 3$                           ; 0425
                                               08    AB  D5 00070        TSTL    8(CCB)                               ;
                                                     04  12 00073        BNEQ    4$                                   ;
                                     50        3C    AB  D0 00075 3$:    MOVL    60(CCB), RABORFAB                    ; 0427
                                     08        08    A0  E8 00079 4$:    BLBS    8(RABORFAB), 5$                      ; 0429
                                     56        08    A0  D0 0007D        MOVL    8(RABORFAB), STS                     ; 0432
                                     55        0C    A0  D0 00081        MOVL    12(RABORFAB), STV                    ; 0433
                                               08    A0  7C 00085 5$:    CLRQ    8(RABORFAB)                          ; 0436
                                                     54  DD 00088 6$:    PUSHL   COND_VAL                             ; 0444
                                     7E        C6    AB  32 0008A        CVTWL   -58(CCB), -(SP)                      ;
                                                     55  DD 0008E        PUSHL   STV                                  ;
                                                     56  DD 00090        PUSHL   STS                                  ;
       7E            54              0C              03  EF 00092        EXTZV   #3, #12, COND_VAL, -(SP)             ;
                          00000000G  00              05  FB 00097        CALLS   #5, FOR$$ERRSNS_SAV                  ;
                                 60  AE        F7    AB  9B 0009E        MOVZBW  -9(CCB), FILE_NAME_DSC               ; 0451
                                               62    AE  B4 000A3        CLRW    FILE_NAME_DSC+2                      ; 0452
                                               F7    AB  95 000A6        TSTB    -9(CCB)                              ; 0454
                                                     04  12 000A9        BNEQ    7$                                   ;
                                                     50  D4 000AB        CLRL    R0                                   ;
                                                     04  11 000AD        BRB     8$                                   ;
                                     50        F8    AB  D0 000AF 7$:    MOVL    -8(CCB), R0                          ; 0458
                                 64  AE              50  D0 000B3 8$:    MOVL    R0, FILE_NAME_DSC+4                  ; 0454
                                     83        C6    AB  32 000B7        CVTWL   -58(CCB), (LIST_PTR)+                ; 0464
                                     83        60    AE  9E 000BB        MOVAB   FILE_NAME_DSC, (LIST_PTR)+           ; 0465
                                                     83  D4 000BF        CLRL    (LIST_PTR)+                          ; 0466
                                     83              56  D0 000C1        MOVL    STS, (LIST_PTR)+                     ; 0467
                                     83              55  D0 000C4        MOVL    STV, (LIST_PTR)+                     ; 0468
                                     57              52  D1 000C7 9$:    CMPL    ARGS_PTR, ARG_LIST_END               ; 0471
                                                     05  14 000CA        BGTR    10$                                  ;
                                     83              82  D0 000CC        MOVL    (ARGS_PTR)+, (LIST_PTR)+             ; 0472
                                                     F6  11 000CF        BRB     9$                                   ;
                                 08  BC        10    AE  FA 000D1 10$:   CALLG   SIGNAL_LIST, @SIGNAL_ROUTINE         ; 0478
                                                         04 000D6        RET                                          ; 0485

; Routine Size:  215 bytes,    Routine Base:  _FOR$CODE + 003C
```

```
1 GLOBAL ROUTINE FOR$$SIG_NO_LUB (            ! SIGNAL_STOP FORTRAN error and STOP
1           FORT_ERR_NO,                ! FORTRAN error code 0:120 or 32-bit cond value
1                                       ! VAX-11 error code
1           FORT_LUN,                   ! Optional FORTRAN logical unit number
1           FAB)                        ! Optional FAB address
1           :NOVALUE =                  ! No value returned.
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !         Convert FORTRAN error code number to 32-bit VAX-11 error code.
1 !         The following SIGNAL_STOP arguments are obtained from the
1 !         argument list only since no LUB/ISB/RAB yet:
1 !
1 !                 VAX-11 error code:
1 !                         STS$V_SEVERITY = STS$K_SEVERE
1 !                         STS$V_CODE = FORTRAN error number
1 !                         STS$V_FAC_SP = 1 (facility specific error messages
1 !                         STS$V_FAC_NO = FORTRAN facility no. (FOR$K_FAC_NO)
1 !                 3 = No. of following FAO arguments
1 !                 FORTRAN unit number if present or zero
1 !                 File name string descriptor address or 0 if no FAB
1 !                 User PC of call to library (set to 0 here, rewritten by handler before RESIGNAL)
1 !                 RMS error code from FAB if present
1 !                 System error code from FAB if present
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !         FORT_ERR_NO.rlu.v       FORTRAN error code (0:120) or 32-bit cond value
1 !                                 32-bit VAX-11 error code with LH already set.
1 !         [FORT_LUN.rlu.v]        Optional unit number, 0 used if not present
1 !         [FAB.rbu.ra]            Address of FAB if present
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !         NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !         NONE
1 !
1 ! COMPLETION CODES:
1 !
1 !         NONE
1 !
1 ! SIDE EFFECTS:
1 !
1 !         Converts FORTRAN error code to 32-bit VAX-11 error code and SIGNAL_STOPs.
1 !--
```

```
BEGIN
LOCAL
    VAX_11_COND_VAL: BLOCK[4,BYTE],  ! 32-bit VAX-11 error code
    NAME_DSC : DSC$DESCRIPTOR,       ! File name descriptor
    STS,                             ! RMS error status
    STV;                             ! System error status
MAP
    FORT_ERR_NO: BLOCK[,BYTE],       ! MAKE 32-bit VAX-11 error code
    FAB : REF BLOCK [,BYTE];         ! FAB is address of FAB
BUILTIN
    ACTUALCOUNT;                                        ! Actual no. of parameters

!+
! Convert FORTRAN error code to 32-bit VAX-11 error code unless
! already converted by the caller.  Conversion is done
! by copying FORTRAN error number to code field,
! setting the severity code to SEVERE,
! setting the facility code to FOR$K_FAC_NO,
! and setting the facility specific bit (STS$V_FAC_SP).
!-

VAX_11_COND_VAL = COND_VALUE (.FORT_ERR_NO);

!+
! If FAB argument is present,  retrieve RMS and SYSTEM error codes.
!-

IF ACTUALCOUNT () GTRU 2
THEN
    BEGIN
    STS = (IF .FAB [FAB$L_STS] THEN 0 ELSE .FAB [FAB$L_STS]);
    STV = (IF .FAB [FAB$L_STV] THEN 0 ELSE .FAB [FAB$L_STV]);
    END
ELSE
    BEGIN
    STS = 0;
    STV = 0;
    END;

!+
! Save FORTRAN error #, RMS STS, RMS STV, logical unit number, and VAX-11 condition value.
! If FORT_LUN not present, use 0 (e.g., INVALID ARG TO FORTRAN I/O SYSTEM)
!-

FOR$$ERRSNS_SAV (.FORT_ERR_NO, .STS, .STV,
    (IF ACTUALCOUNT () GTRU 1 THEN .FORT_LUN ELSE 0), .VAX_11_COND_VAL);

!+
! Set up file name descriptor
!-
NAME_DSC [DSC$B_CLASS] = DSC$K_CLASS_S;
NAME_DSC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
IF ACTUALCOUNT () GTRU 2
THEN
    BEGIN
    IF .FAB [FAB$L_NAM] NEQ 0
    THEN
```

```
                BEGIN
                LOCAL
                    NAM : REF BLOCK [,BYTE];            ! NAM block
                NAM = .FAB [FAB$L_NAM];
                IF .NAM [NAM$B_RS[] NEQ 0
                THEN
                    BEGIN
                    NAME_DSC [DSC$W_LENGTH] = .NAM [NAM$B_RSL];
                    NAME_DSC [DSC$A_POINTER] = .NAM [NAM$[_RSA];
                    END
                ELSE IF .NAM [NAM$B_ESL] NEQ 0
                THEN
                    BEGIN
                    NAME_DSC [DSC$W_LENGTH] = .NAM [NAM$B_ESL];
                    NAME_DSC [DSC$A_POINTER] = .NAM [NAM$[_ESA];
                    END
                ELSE
                    BEGIN
                    NAME_DSC [DSC$W_LENGTH] = .FAB [FAB$B_FNS];
                    NAME_DSC [DSC$A_POINTER] = .FAB [FAB$[_FNA];
                    END;
                END
            ELSE
                BEGIN
                NAME_DSC [DSC$W_LENGTH] = .FAB [FAB$B_FNS];
                NAME_DSC [DSC$A_POINTER] = .FAB [FAB$[_FNA];
                END;
            END
        ELSE
            BEGIN
            NAME_DSC [DSC$W_LENGTH] = 0;
            NAME_DSC [DSC$A_POINTER] = 0;
            END;

        !+
        ! Call LIB$STOP to SIGNAL_STOP the error
        ! Order of args is same as defined in FPAR.MDL for use with SYS$PUT_MESSAGE
        !-

        LIB$STOP (
            .VAX_11_COND_VAL,           ! 32-bit VAX-11 error code
            K_NO_FAO_SIGARG,            ! no. of FAO arguments following in FORTRAN error message
            .FORT_LUN,                  ! FORTRAN logical unit number
            NAME_DSC,                   ! File name descriptor
            0,                          ! Leave room for user PC to be filled in
                                        ! by FORTRAN specific handler established on user call
            .STS,                       ! RMS error code
            .STV);                      ! SYSTEM error code

        !+
        ! Return
        !-

        RETURN
        END;                            ! End of FOR$$SIG_NO_LUB routine
```

```
                                001C 00000          .ENTRY  FOR$$SIG_NO_LUB, Save R2,R3,R4                       ; 0486
             5E               08  C2 00002          SUBL2   #8, SP                                                ;
                        04    AC  DD 00005          PUSHL   FORT_ERR_NO                                           ; 0555
       0000V CF               01  FB 00008          CALLS   #1, COND_VALUE                                        ;
             54               50  D0 0000D          MOVL    R0, VAX_11_COND_VAL                                   ;
             02               6C  91 00010          CMPB    (AP), #2                                              ; 0561
                              1A  1B 00013          BLEQU   3$                                                    ;
             50         0C    AC  D0 00015          MOVL    FAB, R0                                               ; 0564
             04         08    A0  E9 00019          BLBC    8(R0), 1$                                             ;
                              53  D4 0001D          CLRL    STS                                                   ;
                              04  11 0001F          BRB     2$                                                    ;
             53         08    A0  D0 00021 1$:      MOVL    8(R0), STS                                            ;
             08         0C    A0  E8 00025 2$:      BLBS    12(R0), 4$                                            ; 0565
             52         0C    A0  D0 00029          MOVL    12(R0), STV                                           ;
                              04  11 0002D          BRB     5$                                                    ; 0561
                              53  D4 0002F 3$:      CLRL    STS                                                   ; 0569
                              52  D4 00031 4$:      CLRL    STV                                                   ; 0570
                              54  DD 00033 5$:      PUSHL   VAX_11_COND_VAL                                       ; 0579
             01               6C  91 00035          CMPB    (AP), #1                                              ;
                              05  1B 00038          BLEQU   6$                                                    ;
                        08    AC  DD 0003A          PUSHL   FORT_LUN                                              ;
                              02  11 0003D          BRB     7$                                                    ;
                              7E  D4 0003F 6$:      CLRL    -(SP)                                                 ;
                              52  DD 00041 7$:      PUSHL   STV                                                   ; 0578
                              53  DD 00043          PUSHL   STS                                                   ;
                        04    AC  DD 00045          PUSHL   FORT_ERR_NO                                           ;
   00000000G 00               05  FB 00048          CALLS   #5, FOR$$ERRSNS_SAV                                   ;
          02 AE       010E    8F  B0 0004F          MOVW    #270, NAME_DSC+2                                      ; 0585
             02               6C  91 00055          CMPB    (AP), #2                                              ; 0586
                              38  1B 00058          BLEQU   10$                                                   ;
             51         0C    AC  D0 0005A          MOVL    FAB, R1                                               ; 0589
                        28    A1  D5 0005E          TSTL    40(R1)                                                ;
                              24  13 00061          BEQL    9$                                                    ;
             50         28    A1  D0 00063          MOVL    40(R1), NAM                                           ; 0594
                        03    A0  95 00067          TSTB    3(NAM)                                                ; 0595
                              0B  13 0006A          BEQL    8$                                                    ;
             6E         03    A0  9B 0006C          MOVZBW  3(NAM), NAME_DSC                                      ; 0598
          04 AE         04    A0  D0 00070          MOVL    4(NAM), NAME_DSC+4                                    ; 0599
                              20  11 00075          BRB     11$                                                   ; 0595
                        0B    A0  95 00077 8$:      TSTB    11(NAM)                                               ; 0601
                              0B  13 0007A          BEQL    9$                                                    ;
             6E         0B    A0  9B 0007C          MOVZBW  11(NAM), NAME_DSC                                     ; 0604
          04 AE         0C    A0  D0 00080          MOVL    12(NAM), NAME_DSC+4                                   ; 0605
                              10  11 00085          BRB     11$                                                   ; 0601
             6E         34    A1  9B 00087 9$:      MOVZBW  52(R1), NAME_DSC                                      ; 0615
          04 AE         2C    A1  D0 0008B          MOVL    44(R1), NAME_DSC+4                                    ; 0616
                              05  11 00090          BRB     11$                                                   ; 0586
                              6E  B4 00092 10$:     CLRW    NAME_DSC                                              ; 0621
                        04    AE  D4 00094          CLRL    NAME_DSC+4                                            ; 0622
                              52  DD 00097 11$:     PUSHL   STV                                                   ; 0638
                              53  DD 00099          PUSHL   STS                                                   ; 0637
                              7E  D4 0009B          CLRL    -(SP)                                                 ; 0630
                        0C    AE  9F 0009D          PUSHAB  NAME_DSC                                              ;
                        08    AC  DD 000A0          PUSHL   FORT_LUN                                              ; 0633
```

```
                                      03  DD 000A3         PUSHL   #3                                         ; 0630
                                      54  DD 000A5         PUSHL   VAX_11_COND_VAL                            ; 0631
                    00000000G  00     07  FB 000A7         CALLS   #7, LIB$STOP                               ;
                                          04 000AE         RET                                                ; 0645

; Routine Size:  175 bytes,    Routine Base:  _FOR$CODE + 0113
```

```
ROUTINE COND_VALUE (     ! Internal routine to convert from FORTRAN error #
                          ! to VAX-11 condition value
        FORT_ERR_NO)      ! Value of FORTRAN error # (0:120) or 32-bit cond value
          =               ! Value is 32-bit VAX-11 condition value
!++
! FUNCTIONAL DESCRIPTION:
!
!       Converts from FORTRAN error number to 32-bit VAX-11 condition value
!       complete with proper severity and all other fields set.
!       If already a 32-bit condition value (ie GTRU FOR$K_ERR_MAX),
!       no converions is done.  Instead the FORTRAN error # is FOR$K_NOTFORSPE
!       which has a value of 1 and indicates a non-FORTRAN specific error.
!
! FORMAL PARAMETERS:
!
!       FORT_ERR_NO               ! Value of FORTRAN error # (0:120) or 32-bit cond value
!
! IMPLICIT INPUTS:
!
!       NONE
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       32-bit VAX-11 condition value.
!
! SIDE EFFECTS:
!
!       NONE
!--

    BEGIN
    MAP
        FORT_ERR_NO: BLOCK [4, BYTE];            ! Could be a condition value
    LOCAL
        VAX_11_COND_VAL: BLOCK [4, BYTE];        ! 32-bit VAX-11 error condition value

    !+
    ! Convert FORTRAN error code to 32-bit VAX-11 error code, unless already
    ! a 32-bit condition value (some other facility than FOR$ in LH).
    ! Conversion is done by copying FORTRAN error number to code field,
    ! setting the severity code to SEVERE, except error 63 (OUTPUT CONVERIOSN ERROR)
    ! in which case the severity is set to ERROR.
    ! Thus the user must do something explicit in order to continue
    ! for all errors, except 63 (but it has ***s so error flagged).
    ! Therefore the user will not inadverantly use data which had errors in it.
    ! setting the facility code to FOR$K_FAC_NO,
    ! and setting the facility specific bit (STS$V_FAC_SP).
    !-

    IF .FORT_ERR_NO LEQU FOR$K_MAX_ERR
    THEN
        BEGIN
```

```
;  648          0703  3           VAX_11_COND_VAL = 0;
;  649          0704  4           VAX_11_COND_VAL[STS$V_SEVERITY] = (IF .FORT_ERR_NO EQL FOR$K_OUTCONERR
;  650          0705  4                                               THEN
;  651          0706  4                                                     STS$K_ERROR
;  652          0707  4                                               ELSE
;  653          0708  3                                                     STS$K_SEVERE);
;  654          0709  3           VAX_11_COND_VAL[STS$V_CODE] = .FORT_ERR_NO;
;  655          0710  3           VAX_11_COND_VAL[STS$V_FAC_SP] = 1;
;  656          0711  3           VAX_11_COND_VAL[STS$V_FAC_NO] = FOR$K_FAC_NO;
;  657          0712  3           END
;  658          0713  2       ELSE
;  659          0714  2           VAX_11_COND_VAL = .FORT_ERR_NO;
;  660          0715  2
;  661          0716  2       RETURN .VAX_11_COND_VAL
;  662          0717  1       END;                              ! End of COND_VALUE routine



                                                 0000 00000 COND_VALUE:
                                                                       .WORD    Save nothing                                ; 0646
                        0000005D    8F      04   AC  D1 00002          CMPL     FORT_ERR_NO, #93                            ; 0700
                                                 27  1A 0000A          BGTRU    3$                                          ;
                                                 51  D4 0000C          CLRL     VAX_11_COND_VAL                             ; 0703
                                    3F      04   AC  D1 0000E          CMPL     FORT_ERR_NO, #63                            ; 0704
                                                 05  12 00012          BNEQ     1$                                          ;
                                    50           02  D0 00014          MOVL     #2, R0                                      ;
                                                 03  11 00017          BRB      2$                                          ;
                                    50           04  D0 00019 1$:      MOVL     #4, R0                                      ;
           51            03         00           50  F0 0001C 2$:      INSV     R0, #0, #3, VAX_11_COND_VAL                 ;
           51            0C         03      04   AC  F0 00021          INSV     FORT_ERR_NO, #3, #12, VAX_11_COND_VAL       ; 0709
                                    51    8000   8F  A8 00027          BISW2    #32768, VAX_11_COND_VAL                     ; 0710
           51            0C         10           18  F0 0002C          INSV     #24, #16, #12, VAX_11_COND_VAL              ; 0711
                                                 04  11 00031          BRB      4$                                          ; 0700
                                    51      04   AC  D0 00033 3$:      MOVL     FORT_ERR_NO, VAX_11_COND_VAL                ; 0714
                                    50           51  D0 00037 4$:      MOVL     VAX_11_COND_VAL, R0                         ; 0716
                                                     04 0003A          RET                                                  ; 0717

; Routine Size:  59 bytes,    Routine Base:  _FOR$CODE + 01C2


;  663          0718  1 END                                ! End of module
;  664          0719  0 ELUDOM




;                            PSECT SUMMARY
;
;     Name                         Bytes                      Attributes
;
;  _FOR$CODE                           509  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        27         0            581          00:01.0
;   _$255$DUA28:[FORRTL.OBJ]FORLIB.L32;1          711       186        26             52          00:00.6
;   _$255$DUA28:[FORRTL.OBJ]RTLLIB.L32;1           36         0         0              8          00:00.1



;                                            COMMAND QUALIFIERS

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

; Size:          509 code + 0 data bytes
; Run Time:         00:15.6
; Elapsed Time:     00:39.4
; Lines/CPU Min:      2772
; Lexemes/CPU-Min: 16368
; Memory Used:   144 pages
; Compilation Complete
```

0183 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

FORREWRSU
LIS

FORREADIF
LIS

FORREADIO
LIS

FORRECPRO
LIS

FORSIGNAL
LIS

FORREADKO
LIS

FORREWIND
LIS

FORREADSF
LIS

FORREWRSO
LIS

FORREADSN
LIS

FORREADDU
LIS

FORREADSU
LIS

FORSECNDS
LIS

FORREADIL
LIS

FORREADKF
LIS

FORREADKU
LIS

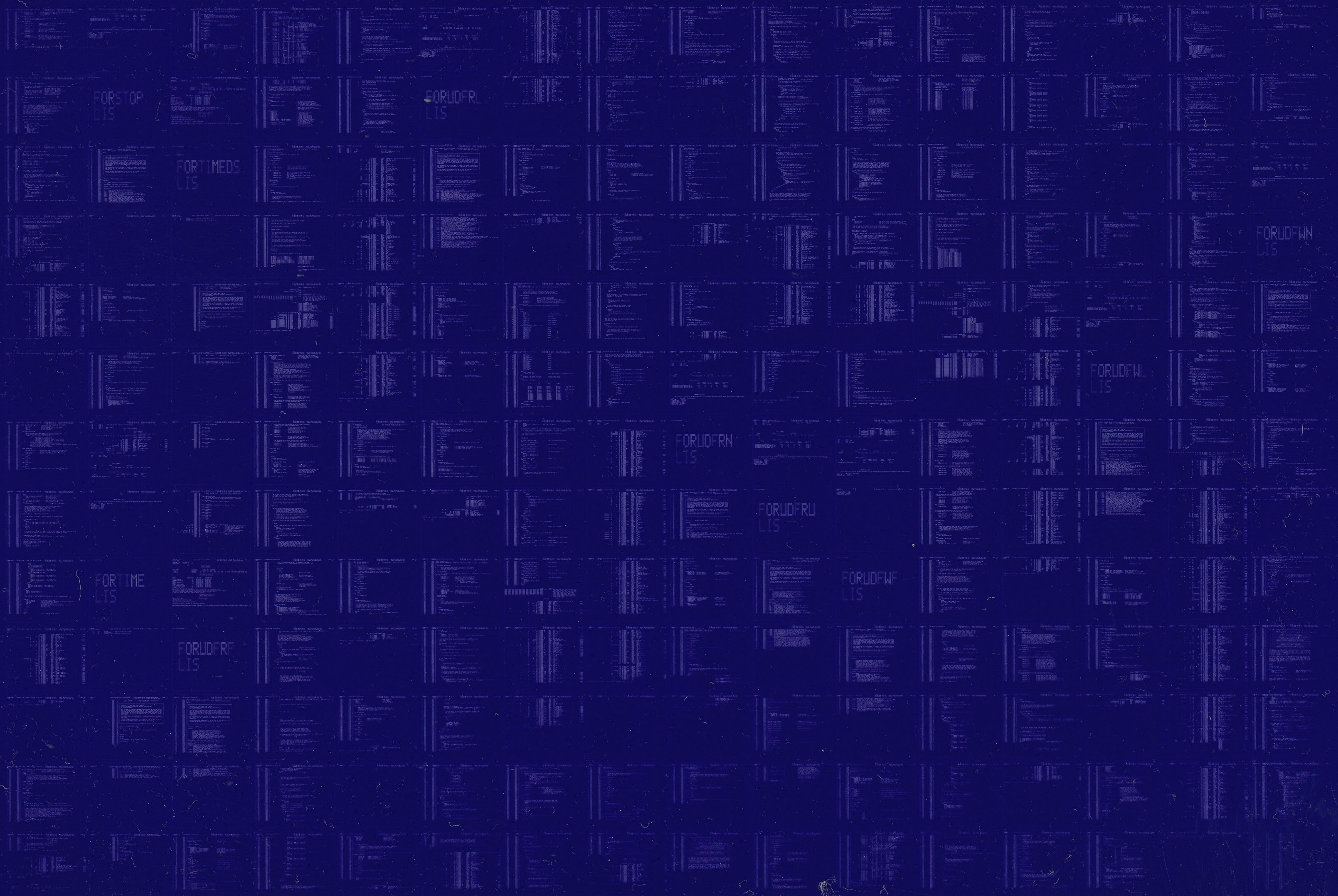
FORREWRSF
LIS

FORREADSL
LIS

FORREADSO
LIS

0184 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY


FORSTOP
LIS

FORUDFRL
LIS

FORTIMEDS
LIS

FORUDFWN
LIS

FORUDFWL
LIS

FORUDFRN
LIS

FORUDFRU
LIS

FORTIME
LIS

FORUDFWF
LIS

FORUDFRF
LIS