```
RRRRRRRRRRR      UUU          UUU  NNN         NNN     000000000     FFFFFFFFFFFFFFF   FFFFFFFFFFFFFFF
RRRRRRRRRRRR     UUU          UUU  NNN         NNN     000000000     FFFFFFFFFFFFFFF   FFFFFFFFFFFFFFF
RRRRRRRRRRRR     UUU          UUU  NNN         NNN     000000000     FFFFFFFFFFFFFFF   FFFFFFFFFFFFFFF
RRR          RRR UUU          UUU  NNN         NNN  000          000 FFF               FFF
RRR          RRR UUU          UUU  NNN         NNN  000          000 FFF               FFF
RRR          RRR UUU          UUU  NNN         NNN  000          000 FFF               FFF
RRR          RRR UUU          UUU  NNNNNN      NNN  000          000 FFF               FFF
RRR          RRR UUU          UUU  NNNNNN      NNN  000          000 FFF               FFF
RRR          RRR UUU          UUU  NNNNNN      NNN  000          000 FFF               FFF
RRRRRRRRRRR      UUU          UUU  NNN   NNN   NNN  000          000 FFFFFFFFFFF       FFFFFFFFFFF
RRRRRRRRRRRR     UUU          UUU  NNN   NNN   NNN  000          000 FFFFFFFFFFF       FFFFFFFFFFF
RRRRRRRRRRRR     UUU          UUU  NNN   NNN   NNN  000          000 FFFFFFFFFFF       FFFFFFFFFFF
RRR   RRR        UUU          UUU  NNN       NNNNNN 000          000 FFF               FFF
RRR   RRR        UUU          UUU  NNN       NNNNNN 000          000 FFF               FFF
RRR   RRR        UUU          UUU  NNN       NNNNNN 000          000 FFF               FFF
RRR      RRR     UUU          UUU  NNN          NNN 000          000 FFF               FFF
RRR      RRR     UUU          UUU  NNN          NNN 000          000 FFF               FFF
RRR      RRR     UUU          UUU  NNN          NNN 000          000 FFF               FFF
RRR         RRR  UUUUUUUUUUUUUUU  NNN          NNN    000000000     FFF               FFF
RRR         RRR  UUUUUUUUUUUUUUU  NNN          NNN    000000000     FFF               FFF
RRR         RRR  UUUUUUUUUUUUUUU  NNN          NNN    000000000     FFF               FFF
RRR         RRR  UUUUUUUUUUUUUUU  NNN          NNN    000000000     FFF               FFF
RRR         RRR  UUUUUUUUUUUUUUU  NNN          NNN    000000000     FFF               FFF
```

```
_$2

Sym
---
NDX
NDX
NUM
NUM
OPE
OUT
PAC
PAC
PAC
PAC
PAC
PAC
PAC
PAD
PAG
PAG
PAG
PAG
PAG
PAG
PAG
PER
PUT
RCO
RIN

RLI
RNO
RNO
RTY
SAV
STR
STR
STR
STR
STR
STR

STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
```

FILEID**ERROR


```
EEEEEEEEEE  RRRRRRRR    RRRRRRRR      000000    RRRRRRRR
EEEEEEEEEE  RRRRRRRR    RRRRRRRR      000000    RRRRRRRR
EE          RR      RR  RR      RR  00      00  RR      RR
EE          RR      RR  RR      RR  00      00  RR      RR
EE          RR      RR  RR      RR  00      00  RR      RR
EE          RR      RR  RR      RR  00      00  RR      RR
EEEEEEEE    RRRRRRRR    RRRRRRRR    00      00  RRRRRRRR
EEEEEEEE    RRRRRRRR    RRRRRRRR    00      00  RRRRRRRR
EE          RR  RR      RR  RR      00      00  RR  RR
EE          RR  RR      RR  RR      00      00  RR  RR
EE          RR    RR    RR    RR    00      00  RR    RR      ....
EE          RR    RR    RR    RR    00      00  RR    RR      ....
EEEEEEEEEE  RR      RR  RR      RR    000000    RR      RR    ....
EEEEEEEEEE  RR      RR  RR      RR    000000    RR      RR    ....


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 %TITLE 'Field (error) message requests from other modules'
;     2          0002  0 MODULE error (  IDENT = 'V04-000'
;     3        P 0003  0                 %BLISS32[, ADDRESSING_MODE (EXTERNAL    = LONG_RELATIVE,
;     4          0004  0                                            NONEXTERNAL = LONG_RELATIVE)]
;     5          0005  0                 ) -
;     6          0006  1 BEGIN
;     7          0007  1 !
;     8          0008  1 !*****************************************************************************
;     9          0009  1 !*                                                                           *
;    10          0010  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;    11          0011  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    12          0012  1 !*  ALL RIGHTS RESERVED.                                                      *
;    13          0013  1 !*                                                                           *
;    14          0014  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
;    15          0015  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
;    16          0016  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
;    17          0017  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
;    18          0018  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
;    19          0019  1 !*  TRANSFERRED.                                                              *
;    20          0020  1 !*                                                                           *
;    21          0021  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
;    22          0022  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT     *
;    23          0023  1 !*  CORPORATION.                                                              *
;    24          0024  1 !*                                                                           *
;    25          0025  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
;    26          0026  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;    27          0027  1 !*                                                                           *
;    28          0028  1 !*                                                                           *
;    29          0029  1 !*****************************************************************************
;    30          0030  1 !
;    31          0031  1 !++
;    32          0032  1 ! FACILITY:      DSR (Digital Standard RUNOFF) / DSRPLUS
;    33          0033  1 !
;    34          0034  1 ! ABSTRACT:
;    35          0035  1 !
;    36          0036  1 !       This module generates messages (generally errors) for RUNOFF.
;    37          0037  1 !       These messages are edited at the time the routine is called to
;    38          0038  1 !       allow inclusion of various information necessary in the
;    39          0039  1 !       diagnostic.  The details of this editing are described below.
;    40          0040  1 !
;    41          0041  1 ! ENVIRONMENT:  Transportable
;    42          0042  1 !
;    43          0043  1 ! AUTHOR:       D. Knight       CREATION DATE:  June 1978
;    44          0044  1 !
;    45          0045  1
```

```
;    47          0046  1 %SBTTL 'Revision History'
;    48          0047  1 !
;    49          0048  1 ! MODIFIED BY:
;    50          0049  1 !
;    51          0050  1 !       014     KAD00014        Keith Dawson    3-May-1983
;    52          0051  1 !               Support suppression on counting new error messages: RNFPC2,
;    53          0052  1 !                 PC3, PC4, PC5, CRU, CR1, CR2).
;    54          0053  1 !
;    55          0054  1 !       013     KAD00013        Keith Dawson    11-Apr-1983
;    56          0055  1 !               Support new termination messages (RNFPC2, PC3, PC4, PC5) for
;    57          0056  1 !                 information written to the .BRN file (and for DSRPLUS cross-
;    58          0057  1 !                 referencing, also read from the input .BRN file). The support
;    59          0058  1 !                 here is the addition of two new %chars:
;    60          0059  1 !
;    61          0060  1 !                       %A ==> spec of old (input) .BRN file
;    62          0061  1 !                       %B ==> spec of new (output) .BRN file
;    63          0062  1 !
;    64          0063  1 !       012     RER00012        Ron Randall     07-Mar-1983
;    65          0064  1 !               Global edit of all modules. Updated module names, idents,
;    66          0065  1 !               copyright dates. Changed require files to BLISS library.
;    67          0066  1 !
;    68          0067  1 !--
;    69          0068  1
```

```
;    71          0069  1 %SBTTL 'Module Level Declarations'
;    72          0070  1
;    73          0071  1 !
;    74          0072  1 ! TABLE OF CONTENTS:
;    75          0073  1 !
;    76          0074  1 FORWARD ROUTINE
;    77          0075  1         erm:    NOVALUE,        ! Error message writer
;    78          0076  1         erma:   NOVALUE,        ! Report error, text of command, & where found
;    79          0077  1         ermb:   NOVALUE,        ! Report error, input text line, & where found
;    80          0078  1         erme:   NOVALUE,        ! Report error extended (w/secondary message)
;    81          0079  1         erml:   NOVALUE,        ! Report error & where in the text it was found
;    82          0080  1         ermn:   NOVALUE,        ! Report error with numeric param & where found
;    83          0081  1         erms:   NOVALUE,        ! Report error with string param & where found
;    84          0082  1         putmsg: NOVALUE,        ! Miscellaneous messages writer
;    85          0083  1         scnmsg,                 ! Do text insertion.
;    86          0084  1         report_secondary_message;! Fetch and report secondary error message
;    87          0085  1
;    88          0086  1 !
;    89          0087  1 ! INCLUDE FILES:
;    90          0088  1 !
;    91          0089  1 LIBRARY 'NXPORT:XPORT';                 ! XPORT Library
;    92          0090  1 REQUIRE 'REQ:RNODEF';                   ! RUNOFF variant definitions
;    93          0221  1
;    94        U 0222  1 %IF DSRPLUS %THEN
;    95        U 0223  1 LIBRARY 'REQ:DPLLIB';                 ! DSRPLUS BLISS Library
;    96          0224  1 %ELSE
;    97          0225  1 LIBRARY 'REQ:DSRLIB';                 ! DSR BLISS Library
;    98          0226  1 %FI
;    99          0227  1
;   100          0228  1 !
;   101          0229  1 ! OWN STORAGE:
;   102          0230  1 !
;   103          0231  1 LITERAL work_buffer_size = 500;
;   104          0232  1
;   105          0233  1 OWN
;   106          0234  1     work_buffer :                               ! Module wide character buffer
;   107          0235  1         VECTOR [CH$ALLOCATION (work_buffer_size)],
;   108        P 0236  1     work_desc : $STR_DESCRIPTOR (               !   & associated string desc.
;   109        P 0237  1                             CLASS=FIXED
;   110        P 0238  1                            ,STRING = (work_buffer_size
;   111          0239  1                                      ,CH$PTR (work_buffer) ) );
;   112          0240  1 OWN
;   113          0241  1     w_buffer1 :                                 ! Module wide character buffer
;   114          0242  1         VECTOR [CH$ALLOCATION (130)],
;   115        P 0243  1     w_desc1 : $STR_DESCRIPTOR (                 !   & associated string desc.
;   116        P 0244  1                             CLASS=FIXED
;   117        P 0245  1                            ,STRING = (130
;   118          0246  1                            ,CH$PTR(w_buffer1)));
;   119          0247  1
;   120          0248  1 !
;   121          0249  1 ! EXTERNAL REFERENCES:
;   122          0250  1 !
;   123          0251  1 EXTERNAL
;   124          0252  1         gca:    gca_definition,
;   125          0253  1         ira:    fixed_string,
;   126          0254  1         irac:   irac_definition,
;   127          0255  1         msgtxt: VECTOR,         ! Vector of CH$PTRs to RUNOFF's messages.
```

```
;  128          0256  1          pagen:  page_definition,
;  129          0257  1          phan:   phan_definition,
;  130          0258  1          rnoiob: REF $XPO_IOB (),! Output file (document being built)
;  131        U 0259  1 %IF DSRPLUS %THEN
;  132        U 0260  1          brniob: $XPO_IOB (),! Input .BRN file
;  133          0261  1 %FI
;  134          0262  1          brnoob: $XPO_IOB (),! Output .BRN file
;  135          0263  1          semcod,                 ! Secondary error message code
;  136          0264  1          ttoiob: $XPO_IOB (),    ! Standard messages (TTY: usually)
;  137          0265  1          tteiob: $XPO_IOB ();    ! Standard error unit (TTY: usually)
;  138          0266  1
;  139          0267  1 EXTERNAL LITERAL                        !Error messages
;  140          0268  1          rnfloc; ! I - on output page %P; on input line %C of page %I of file ''%F''
;  141          0269  1
;  142          0270  1 EXTERNAL ROUTINE
;  143          0271  1          endcmt,                 ! Skip to end of comment
;  144          0272  1          pacbas,                 ! Convert binary to specified base
;  145          0273  1          pacpag,                 ! Convert page number to ASCII
;  146          0274  1          pacstr,                 ! Move ASCII characters.
;  147          0275  1
;  148          0276  1 %IF %BLISS(BLISS32) %THEN
;  149          0277  1          sys$getmsg,             ! System routine to fetch message text
;  150          0278  1          emsg;                   ! Error message handler for VMS
;  151        U 0279  1 %ELSE
;  152        U 0280  1          xpo$xmsg;               ! XPORT's routine to fetch its msg. text
;  153          0281  1 %FI
;  154          0282  1
```

```
;  156          0283  1 %SBTTL 'ERMA -- add DIRECTIVE and location to error message'
;  157          0284  1 GLOBAL ROUTINE erma (error,position) : NOVALUE =
;  158          0285  1
;  159          0286  1 !++
;  160          0287  1 ! FUNCTIONAL DESCRIPTION:
;  161          0288  1 !
;  162          0289  1 !       This routine contains a sequence of code that is commonly used
;  163          0290  1 !       for handling errors.
;  164          0291  1 !
;  165          0292  1 !       If the POSITION parameter is TRUE, it calls ENDCMT to
;  166          0293  1 !       position to the end of the erroneous command.  Then it passes
;  167          0294  1 !       to ERM the specified error message (ERROR), and the start and
;  168          0295  1 !       length of the string to be inserted into the error message.
;  169          0296  1 !       Note that this call on ERM does not depend on the value of POSITION.
;  170          0297  1 !
;  171          0298  1 !       Finally, it calls ERM to output the error message indicating
;  172          0299  1 !       where in the input and output files the error occurred.
;  173          0300  1 !
;  174          0301  1 !       It is by no means necessary to call ERM only via ERMA.  ERMA is
;  175          0302  1 !       simply a collection of commonly used code and is here for the
;  176          0303  1 !       sake of convenience.
;  177          0304  1 !
;  178          0305  1 ! FORMAL PARAMETERS:
;  179          0306  1 !
;  180          0307  1 !       ERROR - Address of desired error diagnostic. Passed to ERM.
;  181          0308  1 !       POSITION - If TRUE, ERMA positions to the end of the
;  182          0309  1 !                  command before issuing the second call on ERM.
;  183          0310  1 !
;  184          0311  1 ! IMPLICIT INPUTS:
;  185          0312  1 !
;  186          0313  1 !       RNFLOC  - An error message stating where error occurred
;  187          0314  1 !       GCA_COM_START - A ch$ptr to the start of the command
;  188          0315  1 !       FS_NEXT (IRA)  - A ch$ptr to the character after the last one
;  189          0316  1 !                        in the command
;  190          0317  1 !
;  191          0318  1 ! IMPLICIT OUTPUTS:     None
;  192          0319  1 !
;  193          0320  1 ! ROUTINE VALUE:
;  194          0321  1 ! COMPLETION CODES:     None
;  195          0322  1 !
;  196          0323  1 ! SIDE EFFECTS:         None
;  197          0324  1 !--
;  198          0325  1
;  199          0326  2     BEGIN
;  200          0327  2
;  201          0328  2     IF  .position
;  202          0329  2     THEN
;  203          0330  2         endcmt ();      ! Position to end of command.
;  204          0331  2
;  205          0332  2     erm (.error, .gca_com_start, CH$DIFF (.fs_next (ira), .gca_com_start));
;  206          0333  2     erm (rnfloc, 0, 0);
;  207          0334  1     END;                ! End of ERMA

;                                                                .TITLE  ERROR Field (error) message requests from other
                                                                         modul
                                                                .IDENT  \V04-000\
```

```
                                                        .PSECT  $OWN$,NOEXE,2

                                                00000 WORK_BUFFER:
                                                        .BLKB   500
                                                01F4  001F4 WORK_DESC:
                                                        .WORD   500
                                                  01  0E  001F6          .BYTE   14, 1                                                   ;
                                            00000000'  001F8          .ADDRESS WORK_BUFFER                                            ;
                                                       001FC W_BUFFER1:
                                                        .BLKB   132
                                                0082  00280 W_DESC1:.WORD   130
                                                  01  0E  00282          .BYTE   14, 1                                                   ;
                                            00000000'  00284          .ADDRESS W_BUFFER1                                              ;

                                                        .EXTRN  GCA, IRA, IRAC, MSGTXT
                                                        .EXTRN  PAGEN, PHAN, RNOIOB
                                                        .EXTRN  BRNOOB, SEMCOD, TTOIOB
                                                        .EXTRN  TTEIOB, RNFLOC, ENDCMT
                                                        .EXTRN  PACBAS, PACPAG, PACSTR
                                                        .EXTRN  SYS$GETMSG, EMSG

                                                        .PSECT  $CODE$,NOWRT,2

                                                000C 00000          .ENTRY  ERMA, Save R2,R3                                        ; 0284
                                    53 00000000V    9E 00002          MOVAB   ERM, R3                                                 ;
                                    52 00000000G EF 9E 00009          MOVAB   GCA, R2                                                 ;
                                    07       08  AC E9 00010          BLBC    POSITION, 1$                                            ; 0328
                          00000000G EF          00 FB 00014          CALLS   #0, ENDCMT                                              ; 0330
                       7E 00000000G EF          62 C3 00C1B 1$:      SUBL3   GCA, IRA+4, -(SP)                                       ; 0332
                                                62 DD 00023          PUSHL   GCA                                                     ;
                                             04 AC DD 00025          PUSHL   ERROR                                                   ;
                                    63          03 FB 00028          CALLS   #3, ERM                                                 ;
                                                7E 7C 0002B          CLRQ    -(SP)                                                   ; 0333
                                       00000000G 8F DD 0002D          PUSHL   #RNFLOC                                                 ;
                                    63          03 FB 00033          CALLS   #3, ERM                                                 ;
                                                   04 00036          RET                                                             ; 0334

; Routine Size:  55 bytes,    Routine Base:  $CODE$ + 0000

;  208          0335  1
```

```
;  210          0336  1 %SBTTL 'ERMB -- add input text & location to error message'
;  211          0337  1 GLOBAL ROUTINE ermb (error,position) : NOVALUE =
;  212          0338  1
;  213          0339  1 !++
;  214          0340  1 ! FUNCTIONAL DESCRIPTION:
;  215          0341  1 !
;  216          0342  1 !       This routine contains a sequence of code that is commonly used
;  217          0343  1 !       for handling errors.
;  218          0344  1 !
;  219          0345  1 !       Processing is like ERMA except that the entire input line is
;  220          0346  1 !       displayed rather than just a command.
;  221          0347  1 !
;  222          0348  1 !       If the POSITION parameter is TRUE, it calls ENDCMT to
;  223          0349  1 !       position to the end of the erroneous command.  It always passes
;  224          0350  1 !       to ERM the specified error message (ERROR), and the start and
;  225          0351  1 !       length of the string to be inserted into the error message.
;  226          0352  1 !       Note that this call on ERM does not depend on the value of POSITION.
;  227          0353  1 !
;  228          0354  1 !       Finally, it calls ERM to output the error message indicating
;  229          0355  1 !       where in the input and output files the error occurred.
;  230          0356  1 !
;  231          0357  1 ! FORMAL PARAMETERS:
;  232          0358  1 !
;  233          0359  1 !       ERROR - Address of desired error diagnostic. Passed to ERM.
;  234          0360  1 !       POSITION - If TRUE, ERMA positions to the end of the
;  235          0361  1 !                  command before issuing the second call on ERM.
;  236          0362  1 !
;  237          0363  1 ! IMPLICIT INPUTS:
;  238          0364  1 !
;  239          0365  1 !       RNFLOC  - An error message stating where error occurred
;  240          0366  1 !       FS_START (IRA)  - A ch$ptr to the start of the input line.
;  241          0367  1 !       FS_MAXSIZE (IRA)        - The total size of the input line.
;  242          0368  1 !
;  243          0369  1 ! IMPLICIT OUTPUTS:     None
;  244          0370  1 !
;  245          0371  1 ! ROUTINE VALUE:
;  246          0372  1 ! COMPLETION CODES:     None
;  247          0373  1 !
;  248          0374  1 ! SIDE EFFECTS:         None
;  249          0375  1 !--
;  250          0376  1
;  251          0377  2     BEGIN
;  252          0378  2
;  253          0379  2     IF .position
;  254          0380  2     THEN
;  255          0381  2         endcmt ();      ! Position to end of command.
;  256          0382  2
;  257          0383  2     erm (.error, .fs_start (ira), .fs_maxsize (ira));
;  258          0384  2     erm (rnfloc, 0, 0);
;  259          0385  1     END;                    ! End of ERMB


                                0004 00000          .ENTRY  ERMB, Save R2                                   ; 0337
                    52 00000000V  EF   9E 00002     MOVAB   ERM, R2                                         ;
```

```
                          07        08   AC  E9 00009          BLBC    POSITION, 1$                                ; 0379
                00000000G EF              00  FB 0000D          CALLS   #0, ENDCMT                                  ; 0381
                   00000000G EF           DD  00014 1$:      PUSHL   IRA+8                                       ; 0383
                   00000000G EF           DD  0001A          PUSHL   IRA                                         ;
                          04 AC           DD  00020          PUSHL   ERROR                                       ;
                62                        03  FB 00023      CALLS   #3, ERM                                     ;
                                          7E  7C 00026      CLRQ    -(SP)                                       ; 0384
                   00000000G 8F           DD  00028          PUSHL   #RNFLOC                                     ;
                62                        03  FB 0002E      CALLS   #3, ERM                                     ;
                                              04 00031      RET                                                 ; 0385

; Routine Size:  50 bytes,    Routine Base:  $CODE$ + 0037

;  260          0386  1
```

```
;  262          0387  1 %SBTTL 'ERME -- report message w/secondary message'
;  263          0388  1 GLOBAL ROUTINE erme (error, arg, len, msgcode) : NOVALUE =
;  264          0389  1
;  265          0390  1 !++
;  266          0391  1 ! FUNCTIONAL DESCRIPTION:
;  267          0392  1 !
;  268          0393  1 !       This routine will report the message as passed and then print a second
;  269          0394  1 !       informational line based on the passed message code (msgcode).
;  270          0395  1 !
;  271          0396  1 ! FORMAL PARAMETERS:
;  272          0397  1 !
;  273          0398  1 !       ERROR -   Address of desired error diagnostic. Passed to ERM.
;  274          0399  1 !       ARG -     A CH$PTR to the string to be output.
;  275          0400  1 !       LEN -     The length of the string.
;  276          0401  1 !       MSGCODE - Error message code for the secondary error message.
;  277          0402  1 !
;  278          0403  1 ! IMPLICIT INPUTS:      None
;  279          0404  1 !
;  280          0405  1 ! IMPLICIT OUTPUTS:     None
;  281          0406  1 !
;  282          0407  1 ! ROUTINE VALUE:
;  283          0408  1 ! COMPLETION CODES:     None
;  284          0409  1 !
;  285          0410  1 ! SIDE EFFECTS:         None
;  286          0411  1 !--
;  287          0412  1
;  288          0413  2     BEGIN
;  289          0414  2     erm (.error, .arg, .len);
;  290          0415  2     report_secondary_message (.msgcode);
;  291          0416  1     END;                    ! End of ERMS


                                                0000 00000      .ENTRY  ERME, Save nothing                              ; 0388
                                    7E      08  AC   7D 00002  MOVQ   ARG, -(SP)                                      ; 0414
                                            04  AC   DD 00006  PUSHL  ERROR
                            00000000V       EF      03  FB 00009  CALLS  #3, ERM
                                            10  AC   DD 00010  PUSHL  MSGCODE                                        ; 0415
                            00000000V       EF      01  FB 00013  CALLS  #1, REPORT_SECONDARY_MESSAGE
                                                    04  0001A  RET                                                    ; 0416

; Routine Size:  27 bytes,    Routine Base:  $CODE$ + 0069

;  292          0417  1
```

```
;  294          0418  1 %SBTTL 'ERML -- add location to error message without additional data'
;  295          0419  1 GLOBAL ROUTINE erml (error) : NOVALUE =
;  296          0420  1
;  297          0421  1 !++
;  298          0422  1 ! FUNCTIONAL DESCRIPTION:
;  299          0423  1 !
;  300          0424  1 !       This routine contains a sequence of code that is commonly used
;  301          0425  1 !       for handling errors.
;  302          0426  1 !
;  303          0427  1 !       ERML is just a shortcut for the following calls:
;  304          0428  1 !               ERM (.ERROR, 0, 0);
;  305          0429  1 !               ERM (RNFLOC, 0, 0);
;  306          0430  1 !       In other words, it saves code.
;  307          0431  1 !
;  308          0432  1 !
;  309          0433  1 ! FORMAL PARAMETERS:
;  310          0434  1 !
;  311          0435  1 !       ERROR - Address of desired error diagnostic. Passed to ERM.
;  312          0436  1 !
;  313          0437  1 ! IMPLICIT INPUTS:
;  314          0438  1 !
;  315          0439  1 !       RNFLOC  - An error message stating where error occurred
;  316          0440  1 !
;  317          0441  1 ! IMPLICIT OUTPUTS:     None
;  318          0442  1 !
;  319          0443  1 ! ROUTINE VALUE:
;  320          0444  1 ! COMPLETION CODES:     None
;  321          0445  1 !
;  322          0446  1 ! SIDE EFFECTS:         None
;  323          0447  1 !--
;  324          0448  1
;  325          0449  2     BEGIN
;  326          0450  2     erm (.error, 0, 0);
;  327          0451  2     erm (rnfloc, 0, 0);
;  328          0452  1     END;                    ! End of ERML


                                                0004 00000          .ENTRY  ERML, Save R2                           ; 0419
                                52 00000000V    EF   9E 00002       MOVAB   ERM, R2
                                                7E   7C 00009       CLRQ    -(SP)                                   ; 0450
                                          04    AC   DD 0000B       PUSHL   ERROR
                                62              03   FB 0000E       CALLS   #3, ERM
                                                7E   7C 00011       CLRQ    -(SP)                                   ; 0451
                                   00000000G    8F   DD 00013       PUSHL   #RNFLOC
                                62              03   FB 00019       CALLS   #3, ERM
                                                     04 0001C       RET                                             ; 0452

; Routine Size:  29 bytes,    Routine Base:  $CODE$ + 0084

;  329          0'53  1
```

```
;   331          0454  1 %SBTTL 'ERMN -- add location to error message w/numeric data'
;   332          0455  1 GLOBAL ROUTINE ermn (error, arg) : NOVALUE =
;   333          0456  1
;   334          0457  1 !++
;   335          0458  1 ! FUNCTIONAL DESCRIPTION:
;   336          0459  1 !
;   337          0460  1 !       This routine contains a sequence of code that is commonly used
;   338          0461  1 !       for handling errors.
;   339          0462  1 !
;   340          0463  1 !       The code is straightforward.  It is mainly a code-saver.
;   341          0464  1 !
;   342          0465  1 !
;   343          0466  1 ! FORMAL PARAMETERS:
;   344          0467  1 !
;   345          0468  1 !       ERROR - Address of desired error diagnostic. Passed to ERM.
;   346          0469  1 !       ARG -   A number to be output.
;   347          0470  1 !
;   348          0471  1 ! IMPLICIT INPUTS:
;   349          0472  1 !
;   350          0473  1 !       RNFLOC  - An error message stating where error occurred
;   351          0474  1 !
;   352          0475  1 ! IMPLICIT OUTPUTS:     None
;   353          0476  1 !
;   354          0477  1 ! ROUTINE VALUE:
;   355          0478  1 ! COMPLETION CODES:     None
;   356          0479  1 !
;   357          0480  1 ! SIDE EFFECTS:         None
;   358          0481  1 !--
;   359          0482  1
;   360          0483  2     BEGIN
;   361          0484  2     erm (.error, .arg, 0);
;   362          0485  2     erm (rnfloc, 0, 0);
;   363          0486  1     END;                    ! End of ERMN


                                                0004 00000          .ENTRY  ERMN, Save R2                                   ; 0455
                                52 00000000V    EF   9E 00002       MOVAB   ERM, R2                                         ;
                                                7E   D4 00009       CLRL    -(SP)                                           ; 0484
                                7E        04    AC   7D 0000B       MOVQ    ERROR, -(SP)                                    ;
                                62              03   FB 0000F       CALLS   #3, ERM                                         ;
                                                7E   7C 00012       CLRQ    -(SP)                                           ; 0485
                                   00000000G    8F   DD 00014       PUSHL   #RNFLOC                                         ;
                                62              03   FB 0001A       CALLS   #3, ERM                                         ;
                                                     04 0001D       RET                                                     ; 0486

; Routine Size:  30 bytes,    Routine Base:  $CODE$ + 00A1

;   364          0487  1
```

```
;  366          0488  1 %SBTTL 'ERMS -- add location to error message w/string data'
;  367          0489  1 GLOBAL ROUTINE erms (error, arg, len) : NOVALUE =
;  368          0490  1
;  369          0491  1 !++
;  370          0492  1 ! FUNCTIONAL DESCRIPTION:
;  371          0493  1 !
;  372          0494  1 !       This routine contains a sequence of code that is commonly used
;  373          0495  1 !       for handling errors.
;  374          0496  1 !
;  375          0497  1 !       The code is straightforward.  It is mainly a code-saver.
;  376          0498  1 !
;  377          0499  1 !                                                                                              ;
;  378          0500  1 ! FORMAL PARAMETERS:
;  379          0501  1 !
;  380          0502  1 !       ERROR - Address of desired error diagnostic. Passed to ERM.                            ;
;  381          0503  1 !       ARG -   A CH$PTR to the string to be output.
;  382          0504  1 !       LEN -   The length of the string.
;  383          0505  1 !
;  384          0506  1 ! IMPLICIT INPUTS:
;  385          0507  1 !
;  386          0508  1 !       RNFLOC  - An error message stating where error occurred
;  387          0509  1 !
;  388          0510  1 ! IMPLICIT OUTPUTS:     None
;  389          0511  1 !
;  390          0512  1 ! ROUTINE VALUE:
;  391          0513  1 ! COMPLETION CODES:     None
;  392          0514  1 !
;  393          0515  1 ! SIDE EFFECTS:         None
;  394          0516  1 !--
;  395          0517  1
;  396          0518  2     BEGIN
;  397          0519  2     erm (.error, .arg, .len);
;  398          0520  2     erm (rnfloc, 0, 0);
;  399          0521  1     END;                    ! End of ERMS


                                        0004 00000         .ENTRY  ERMS, Save R2                                   ; 0489
                            52 00000000V EF   9E 00002     MOVAB   ERM, R2                                         ;
                            7E       08 AC   7D 00009      MOVQ    ARG, -(SP)                                      ; 0519
                                     04 AC   DD 0000D      PUSHL   ERROR                                           ;
                            62          03   FB 00010      CALLS   #3, ERM                                         ;
                                        7E   7C 00013      CLRQ    -(SP)                                           ; 0520
                                00000000G   8F DD 00015    PUSHL   #RNFLOC                                         ;
                            62          03   FB 0001B      CALLS   #3, ERM                                         ;
                                        04 0001E           RET                                                     ; 0521

; Routine Size:  31 bytes,    Routine Base:  $CODE$ + 00BF

;  400          0522  1
```

```
;  402          0523  1 %SBTTL 'ERM -- funnel message to appropriate error output mechinism'
;  403          0524  1 GLOBAL ROUTINE erm (error, arg, len) : NOVALUE =
;  404          0525  1
;  405          0526  1 !++
;  406          0527  1 ! FUNCTIONAL DESCRIPTION:
;  407          0528  1 !
;  408          0529  1 !       This routine generates and prints the requested error diagnostic.
;  409          0530  1 !       The first routine argument points to an error string consisting of
;  410          0531  1 !       a string intermixed with special formatting flags which cause special
;  411          0532  1 !       actions to happen to the string when a flag is encountered.
;  412          0533  1 !       The routine SCNMSG is called to expand the error message
;  413          0534  1 !       before it is output.
;  414          0535  1 !
;  415          0536  1 ! FORMAL PARAMETERS:
;  416          0537  1 !
;  417          0538  1 !       ERROR - Pointer to the desired error diagnostic.
;  418          0539  1 !       ARG - Contains a value if it is to be used with %N;
;  419          0540  1 !             contains a pointer to a string if it is to be used with %S.
;  420          0541  1 !       LEN - Unused with %N, Contains the string length for %S.
;  421          0542  1 !
;  422          0543  1 !       All parameters are "pass through" parameters for SCNMSG.
;  423          0544  1 !
;  424          0545  1 ! IMPLICIT INPUTS:      None
;  425          0546  1 !
;  426          0547  1 ! IMPLICIT OUTPUTS:
;  427          0548  1 !
;  428          0549  1 !       GCA_FEHLER      - Set to TRUE to indicate an error occurred
;  429          0550  1 !       GCA_ERRCNT      - Count of lines of error messages
;  430          0551  1 !
;  431          0552  1 ! ROUTINE VALUE:
;  432          0553  1 ! COMPLETION CODES:     None
;  433          0554  1 !
;  434          0555  1 ! SIDE EFFECTS:         None
;  435          0556  1 !--
;  436          0557  1
;  437          0558  2     BEGIN
;  438          0559  2     EXTERNAL LITERAL                    !Error messages previously undeclaired
;  439          0560  2         rnfbak, ! I - See command on input line %C of page %I of file "%S"
;  440          0561  2         rnferd, ! I - DSR(PLUS) Version %V: %N diagnostic message%X reported
;  441          0562  2         rnfmrc, ! W - Another %N crossed margin or bad right indent attempt%X detected and accumulated.  Now
;  442          0563  2         rnfned, ! I - DSR(PLUS) Version %V: No errors detected
;  443          0564  2         rnfnic, ! W - Another %N negative indent%X detected and accumulated. Now being reported
;  444          0565  2         rnfpc1, ! I - Illegal message:  rnfpc1
;  445          0566  2         rnfpct, ! I - %N page%X written to "%O"
;  446          0567  2         rnfstr, ! I - "%S"
;  447        U 0568  2 %IF DSRPLUS %THEN
;  448        U 0569  2         rnfpc4, ! %N cross-reference record%X written to %B
;  449        U 0570  2         rnfpc5, ! %N cross-reference record%X read from %A
;  450        U 0571  2         rnfcru, ! One or more forward cross-references could not be resolved
;  451        U 0572  2         rnfcr1, ! Run DSRPLUS again if you need them correct
;  452        U 0573  2         rnfcr2, ! Or run DSRPLUS/DEBUG to see which one(s) changed
;  453          0574  2 %FI
;  454          0575  2         rnfpc2, ! %N indexing record%X written to %B
;  455          0576  2         rnfpc3; ! %N table-of-contents record%X written to %B
;  456          0577  2
;  457          0578  2     LITERAL     e_number_mask   = %X'7FF8';
;  458          0579  2
```

```
;  459          0580  2      LOCAL
;  460          0581  2          line_cnt,                                   !Size of output line
;  461          0582  2          e_number,                                   !error number
;  462          0583  2          status;                                     .Status code returned by XPORT
;  463          0584  2
;  464          0585  2  %IF %BLISS(BLISS32) %THEN
;  465          0586  2      ! Initialize message string descriptor:
;  466          0587  2      work_desc [str$b_dtype] = str$k_dtype_t;     ! ASCII text (8-bit)
;  467          0588  2      work_desc [str$b_class] = str$k_class_f;     ! Scalar, String Descriptor
;  468          0589  2      work_desc [str$a_pointer] = work_buffer [0];        ! First byte of char. buffer
;  469          0590  2  %FI
;  470          0591  2
;  471          0592  2      e_number = (.error AND e_number_mask)/8;
;  472          0593  2
;  473          0594  2      !Count every real error message
;  474          0595  3      IF (     .error NEQ rnfbak
;  475          0596  3          AND .error NEQ rnferd
;  476          0597  3          AND .error NEQ rnfloc
;  477          0598  3          AND .error NEQ rnfmrc    ! The parameter passed will be added in later.
;  478          0599  3          AND .error NEQ rnfnic    ! The parameter passed will be added in later.
;  479          0600  3          AND .error NEQ rnfned
;  480          0601  3          AND .error NEQ rnfpct
;  481          0602  3          AND .error NEQ rnfpc1
;  482          0603  3          AND .error NEQ rnfpc2
;  483          0604  3          AND .error NEQ rnfpc3
;  484        U 0605  3  %IF DSRPLUS %THEN
;  485        U 0606  3          AND .error NEQ rnfpc4
;  486        U 0607  3          AND .error NEQ rnfpc5
;  487        U 0608  3          AND .error NEQ rnfcru
;  488        U 0609  3          AND .error NEQ rnfcr1
;  489        U 0610  3          AND .error NEQ rnfcr2
;  490          0611  3  %FI
;  491          0612  3          AND .error NEQ rnfstr  )
;  492          0613  2          THEN
;  493          0614  3              BEGIN
;  494          0615  3              gca_fehler  = true;
;  495          0616  3              gca_errcnt  = .gca_errcnt + 1;
;  496          0617  2              END;
;  497          0618  2
;  498          0619  3      IF (     .error EQL rnfmrc    ! These two messages display an accumulated
;  499          0620  3          OR  .error EQL rnfnic )  ! count of errors previously unreported.
;  500          0621  2          THEN                     !   Add their count to total counter.
;  501          0622  2              gca_errcnt  = .gca_errcnt + .arg;
;  502          0623  2
;  503          0624  2      line_cnt = scnmsg (CH$PTR (work_buffer), .msgtxt [.e_number], .arg, .len);
;  504          0625  2
;  505          0626  2  %IF %BLISS(BLISS32) %THEN
;  506          0627  2      work_desc [str$h_length] = .line_cnt;        ! Put message length into descriptor
;  507          0628  2  %FI
;  508          0629  2
;  509          0630  2      !The line is now packed correctly, so output it to the requested places.
;  510          0631  2      CASE .gca_err_dir FROM report_err_none TO report_err_both OF
;  511          0632  2          SET
;  512          0633  2          [report_err_none]:        0;    !Don't bother to output the message
;  513          0634  2
;  514          0635  2          [report_err_file]:
;  515          0636  3                  BEGIN           !Report error in output file
```

```
;  516          0637  3
;  517        U 0638  3 %IF NOT %BLISS(BLISS32) %THEN
;  518        U 0639  3                   status = $XPO_PUT (IOB    = .rnoiob
;  519        U 0640  3                                     ,string = (.line_cnt, CH$PTR (work_buffer)) );
;  520        U 0641  3
;  521        U 0642  3                   ! Add carriage control information to the end of error message
;  522        U 0643  3                   status = $XPO_PUT  (IOB = .rnoiob
;  523        U 0644  3                                      ,STRING = (2, CH$PTR (UPLIT (%STRING (%CHAR (
;  524        U 0645  3                                                                %O'15' ,%O'12'))))) );
;  525          0646  3 %ELSE
;  526          0647  3                   status = emsg (.error, work_desc, true);
;  527          0648  3 %FI
;  528          0649  3
;  529          0650  2                   END;
;  530          0651  2
;  531          0652  2           [report_err_std]:
;  532          0653  2
;  533          0654  3                   BEGIN           ! Report error on standard error log
;  534          0655  3
;  535        U 0656  3 %IF NOT %BLISS(BLISS32) %THEN
;  536        U 0657  3                   status = $XPO_PUT( IOB = tteiob
;  537        U 0658  3                                     ,STRING = (.line_cnt, CH$PTR (work_buffer)) );
;  538          0659  3 %ELSE
;  539          0660  3                   status = emsg (.error, work_desc, false);
;  540          0661  3 %FI
;  541          0662  3
;  542          0663  2                   END;
;  543          0664  2
;  544          0665  2           [report_err_both]:
;  545          0666  3                   BEGIN           ! Report error both places -- output file first
;  546          0667  3
;  547        U 0668  3 %IF NOT %BLISS(BLISS32) %THEN
;  548        U 0669  3                   status = $XPO_PUT (IOB = .rnoiob
;  549        U 0670  3                                     ,STRING = (.line_cnt, CH$PTR (work_buffer)));
;  550        U 0671  3
;  551        U 0672  3                   !Add carriage control information to the end of error message
;  552        U 0673  3                   status = $XPO_PUT  (IOB = .rnoiob
;  553        U 0674  3                                      ,STRING = (2, CH$PTR (UPLIT (%STRING (%CHAR(
;  554        U 0675  3                                                                %O'15', %O'12'))))));
;  555        U 0676  3                   !Report error on standard error log
;  556        U 0677  3                   status = $XPO_PUT (IOB = tteiob
;  557        U 0678  3                                     ,STRING = (.line_cnt, CH$PTR (work_buffer)));
;  558          0679  3 %ELSE
;  559          0680  3                   status = emsg (.error, work_desc, true);
;  560          0681  3 %FI
;  561          0682  3
;  562          0683  2                   END;
;  563          0684  2
;  564          0685  2               TES;
;  565          0686  2
;  566          0687  1     END;                        !End of ERM


                                                                .EXTRN  RNFBAK, RNFERD, RNFMRC
                                                                .EXTRN  RNFNED, RNFNIC, RNFPC1
                                                                .EXTRN  RNFPCT, RNFSTR, RNFPC2
```

```
                                                        .EXTRN  RNFPC3

                                                 007C 00000     .ENTRY  ERM, Save R2,R3,R4,R5,R6                        ; 0524
                                    56 00000000G   8F   D0 00002    MOVL    #RNFNIL, R6
                                    55 00000000G   8F   D0 00009    MOVL    #RNFMRC, R5
                                    54 00000000G   EF   9E 00010    MOVAB   GCA+196, R4
                                    53 00000000'   EF   9E 00017    MOVAB   WORK_DESC, R3
                                 02 A3     010E    8F   B0 0001E    MOVW    #270, WORK_DESC+2                       ; 0587
                                 04 A3     FE0C    C3   9E 00024    MOVAB   WORK_BUFFER, WORK_DESC+4                ; 0589
                                    52       04    AC   D0 0002A    MOVL    ERROR, R2                               ; 0592
                              50    52 FFFF8007    8F   CB 0002E    BICL3   #-32761, R2, R0
                                    50             08   C6 00036    DIVL2   #8, E_NUMBER
                           00000000G   8F         52   D1 00039    CMPL    R2, #RNFBAK                             ; 0595
                                                58   13 00040    BEQL    1$
                           00000000G   8F         52   D1 00042    CMPL    R2, #RNFERD                             ; 0596
                                                4F   13 00049    BEQL    1$
                           00000000G   8F         52   D1 0004B    CMPL    R2, #RNFLOC                             ; 0597
                                                46   13 00052    BEQL    1$
                                    55          52   D1 00054    CMPL    R2, R5                                  ; 0598
                                                41   13 00057    BEQL    1$
                                    56          52   D1 00059    CMPL    R2, R6                                  ; 0599
                                                3C   13 0005C    BEQL    1$
                           00000000G   8F         52   D1 0005E    CMPL    R2, #RNFNED                             ; 0600
                                                33   13 00065    BEQL    1$
                           00000000G   8F         52   D1 00067    CMPL    R2, #RNFPCT                             ; 0601
                                                2A   13 0006E    BEQL    1$
                           00000000G   8F         52   D1 00070    CMPL    R2, #RNFPC1                             ; 0602
                                                21   13 00077    BEQL    1$
                           00000000G   8F         52   D1 00079    CMPL    R2, #RNFPC2                             ; 0603
                                                18   13 00080    BEQL    1$
                           00000000G   8F         52   D1 00082    CMPL    R2, #RNFPC3                             ; 0604
                                                0F   13 00089    BEQL    1$
                           00000000G   8F         52   D1 0008B    CMPL    R2, #RNFSTR                             ; 0612
                                                06   13 00092    BEQL    1$
                                 FC    A4         01   D0 00094    MOVL    #1, GCA+192                             ; 0615
                                                64   D6 00098    INCL    GCA+196                                 ; 0616
                                    55          52   D1 0009A 1$: CMPL    R2, R5                                  ; 0619
                                                05   13 0009D    BEQL    2$
                                    56          52   D1 0009F    CMPL    R2, R6                                  ; 0620
                                                04   12 000A2    BNEQ    3$
                                    64       08    AC   C0 000A4 2$: ADDL2   ARG, GCA+196                           ; 0622
                                    7E       08    AC   7D 000A8 3$: MOVQ    ARG, -(SP)                              ; 0624
                                       00000000GEF   40   DD 000AC    PUSHL   MSGTXT[E_NUMBER]
                                          FE0C    C3   9F 000B3    PUSHAB  WORK_BUFFER
                           00000000V      EF         04   FB 000B7    CALLS   #4, SCNMSG
                                    63             50   B0 000BE    MOVW    LINE_CNT, WORK_DESC                     ; 0627
                              03    00       04    A4   CF 000C1    CASEL   GCA+200, #0, #3                         ; 0631
                        000D  0009    000D     0018      000C6 4$: .WORD   8$-4$,-
                                                                             6$-4$,-
                                                                             5$-4$,-
                                                                             6$-4$
                                                 04 000CE       RET
                                                7E   D4 000CF 5$: CLRL    -(SP)                                   ; 0660
                                                02   11 000D1    BRB     7$
                                                01   DD 000D3 6$: PUSHL   #1                                      ; 0680
                                                0C   BB 000D5 7$: PUSHR   #^M<R2,R3>
                           00000000G   EF         03   FB 000D7    CALLS   #3, EMSG
```

```
                                        04 000DE 8$:     RET                                                    ; 0687

; Routine Size:  223 bytes,    Routine Base:  $CODE$ + 00DE

;  567          0688  1
```

```
;  569          0689  1 %SBTTL 'PUTMSG -- funnel message to appropriate output mechinism'
;  570          0690  1 GLOBAL ROUTINE putmsg (message, arg, len) : NOVALUE =
;  571          0691  1
;  572          0692  1 !++
;  573          0693  1 ! FUNCTIONAL DESCRIPTION:
;  574          0694  1 !
;  575          0695  1 !       This routine generates and prints the requested message.
;  576          0696  1 !       The first routine argument points to a string consisting of
;  577          0697  1 !       a string intermixed with special formatting flags which cause special
;  578          0698  1 !       actions to happen to the string when a flag is encountered.
;  579          0699  1 !       The routine SCNMSG is called to expand the message
;  580          0700  1 !       before it is output.
;  581          0701  1 !
;  582          0702  1 ! FORMAL PARAMETERS:
;  583          0703  1 !
;  584          0704  1 !       MESSAGE - Pointer to the desired message.
;  585          0705  1 !       ARG - Contains a value, if it is to be used with %N,
;  586          0706  1 !             contains a pointer to a string if it is to be used with %S.
;  587          0707  1 !       LEN - Unused with %N, Contains the string length for %S.
;  588          0708  1 !
;  589          0709  1 !       All parameters are "pass through" parameters for SCNMSG.
;  590          0710  1 !
;  591          0711  1 ! IMPLICIT INPUTS:      None
;  592          0712  1 !
;  593          0713  1 ! IMPLICIT OUTPUTS:     None
;  594          0714  1 !
;  595          0715  1 ! ROUTINE VALUE:
;  596          0716  1 ! COMPLETION CODES:     None
;  597          0717  1 !
;  598          0718  1 ! SIDE EFFECTS:         None
;  599          0719  1 !--
;  600          0720  1
;  601          0721  2     BEGIN
;  602          0722  2     LITERAL
;  603          0723  2         e_number_mask    = %X'7FF8';
;  604          0724  2     LOCAL
;  605          0725  2         line_cnt,                                   !Size of output line
;  606          0726  2         e_number,                                   !error number
;  607          0727  2         status;                                     !Status code returned by XPORT
;  608          0728  2
;  609          0729  2 %IF %BLISS(BLISS32) %THEN
;  610          0730  2     ! Initialize message string descriptor:
;  611          0731  2
;  612          0732  2     work_desc [str$b_dtype] = str$k_dtype_t;     ! ASCII text (8-bit)
;  613          0733  2     work_desc [str$b_class] = str$k_class_f;     ! Scalar, String Descriptor
;  614          0734  2     work_desc [str$a_pointer] = work_buffer [0];        ! First byte of char. buffer
;  615          0735  2 %FI
;  616          0736  2
;  617          0737  2     e_number = (.message AND e_number_mask) / 8;
;  618          0738  2     line_cnt = scnmsg (CH$PTR (work_buffer), .msgtxt [.e_number], .arg, .len);
;  619          0739  2
;  620          0740  2 %IF %BLISS(BLISS32) %THEN
;  621          0741  2     work_desc [str$h_length] = .line_cnt;       ! Put message length into descriptor
;  622          0742  2 %FI
;  623          0743  2
;  624          0744  2     !The line is now packed .orrectly, so output it to the terminal
;  625          0745  2
```

```
;  626          U 0746  2 %IF NOT %BLISS(BLISS32) %THEN
;  627          U 0747  2     status = $XPO_PUT (IOB = ttoiob     !Standard message device (usually TTY:)
;  628          U 0748  2                       ,STRING = (.line_cnt, CH$PTR (work_buffer)));
;  629            0749  2 %ELSE
;  630            0750  2     status =    emsg (.message, work_desc, false);
;  631            0751  2 %FI
;  632            0752  2
;  633            0753  1     END;                        !End of PUTMSG


                                                0004 00000         .ENTRY  PUTMSG, Save R2                                 ; 0690
                                52 00000000'    EF   9E 00002      MOVAB   WORK_DESC, R2
                             02 A2      010E    8F   B0 00009      MOVW    #270, WORK_DESC+2                               ; 0732
                             04 A2      FE0C    C2   9E 0000F      MOVAB   WORK_BUFFER, WORK_DESC+4                        ; 0734
                        50   04 AC FFFF8007    8F   CB 00015      BICL3   #-32761, MESSAGE, R0                            ; 0737
                                50              08   C6 0001E      DIVL2   #8, E_NUMBER
                                7E        08    AC   7D 00021      MOVQ    ARG, -(SP)                                      ; 0738
                                   00000000GEF  40   DD 00025      PUSHL   MSGTXT[E_NUMBER]
                                      FE0C      C2   9F 0002C      PUSHAB  WORK_BUFFER
                                00000000V EF    04   FB 00030      CALLS   #4, $CNMSG
                                62              50   B0 00037      MOVW    LINE_CNT, WORK_DESC                             ; 0741
                                                7E   D4 0003A      CLRL    -(SP)                                           ; 0750
                                                52   DD 0003C      PUSHL   R2
                                             04 AC   DD 0003E      PUSHL   MESSAGE
                                00000000G  EF   03   FB 00041      CALLS   #3, EMSG
                                                04   00048        RET                                                     ; 0753

; Routine Size:  73 bytes,    Routine Base:  $CODE$ + 01BD

;  634          0754  1
```

```
;  636     0755  1 %SBTTL 'SCNMSG -- fill in missing arguments in message string'
;  637     0756  1 ROUTINE scnmsg (line, message, arg, len) =
;  638     0757  1
;  639     0758  1 !++
;  640     0759  1 ! FUNCTIONAL DESCRIPTION:
;  641     0760  1 !
;  642     0761  1 !       This routine expands the specified message. The first routine
;  643     0762  1 !       argument is a ch$ptr to a work area where the expanded message is
;  644     0763  1 !       written to. The second routine argument points to a string consisting
;  645     0764  1 !       of a string intermixed with special formatting flags which cause special
;  646     0765  1 !       actions to happen to the string when a flag is encountered. The legal
;  647     0766  1 !       flags in the string are:
;  648     0767  1 !
;  649     0768  1 !               %S - Using the second and third call arguments place the
;  650     0769  1 !                    specified string at this point.
;  651     0770  1 !               %N - Using the second argument only, place the specified
;  652     0771  1 !                    decimal number at this point.
;  653     0772  1 !               %P - Insert the Current Output Page.
;  654     0773  1 !               %L - Insert the Current Output Line.
;  655     0774  1 !               %I - Insert the input page number.
;  656     0775  1 !               %C - Insert the input sequence number.
;  657     0776  1 !               %F - Insert the name of the current input file.
;  658     0777  1 !               %O - Insert the name of the output file.
;  659     0778  1 !               %A - Insert the name of the old (input) .BRN file.
;  660     0779  1 !               %B - Insert the name of the new (output) .BRN file.
;  661     0780  1 !               %V - Insert the program version number
;  662     0781  1 !               %X - Insert "s" if %N is other then 1.
;  663     0782  1 !
;  664     0783  1 !       More than one of the above arguments can occur in a message,
;  665     0784  1 !       except that %S and %N are mutually exclusive and cannot occur
;  666     0785  1 !       in the same message.
;  667     0786  1 !
;  668     0787  1 !       An illegal flag will be interpreted as plaintext.
;  669     0788  1 !
;  670     0789  1 ! FORMAL PARAMETERS:
;  671     0790  1 !
;  672     0791  1 !       LINE - Ch$ptr to where the message is to be built up
;  673     0792  1 !       MESSAGE - Pointer to the desired message (unexpanded).
;  674     0793  1 !       ARG - Contains a value, if it is to be used with %N,
;  675     0794  1 !             contains a pointer to a string if it is to be used with %S.
;  676     0795  1 !       LEN - Unused with %N, Contains the string length for %S.
;  677     0796  1 !
;  678     0797  1 ! IMPLICIT INPUTS:
;  679     0798  1 !
;  680     0799  1 !       PAGEN           - Output Page number
;  681     0800  1 !       PHAN_LINES_TP   - Output Line number
;  682     0801  1 !       IRAC_IPAGEN     - Input page number
;  683     0802  1 !       IRAC_ISEQN      - input line count
;  684     0803  1 !       IRAC_FSPECP     - Pointer to Input file name string
;  685     0804  1 !       IRAC_FSPECC     - Length of Input file name string
;  686     0805  1 !
;  687     0806  1 ! IMPLICIT OUTPUTS:     None
;  688     0807  1 !
;  689     0808  1 ! ROUTINE VALUE:
;  690     0809  1 !
;  691     0810  1 !       Returns the number of characters in the expanded message.
;  692     0811  1 !
```

```
;  693          0812  1 ! SIDE EFFECTS:          None
;  694          0813  1 !--
;  695          0814  1
;  696          0815  2     BEGIN
;  697          0816  2     LOCAL
;  698          0817  2         line_cnt,                                   !Size of output line
;  699          0818  2         line_ptr,                                   !String pointer to output line
;  700          0819  2         strg_cnt,                                   !Size of input line
;  701          0820  2         strg_ptr;                                   !String pointer to input line
;  702          0821  2
;  703          0822  2     !Set up for processing
;  704          0823  2     line_ptr=.line;
;  705          0824  2     line_cnt=0;
;  706          0825  2     strg_ptr=.message;
;  707          0826  2
;  708          0827  2     !Get input line size (length of orginial error message string)
;  709          0828  2     strg_cnt = CH$RCHAR_A (strg_ptr);
;  710          0829  2
;  711          0830  2     !Now process the entire input string
;  712          0831  2     INCR i FROM 1 TO .strg_cnt DO
;  713          0832  3         BEGIN
;  714          0833  3         LOCAL
;  715          0834  3             character;
;  716          0835  3
;  717          0836  3         character = CH$RCHAR_A (strg_ptr);
;  718          0837  3
;  719          0838  3         IF .character NEQ %C'%'
;  720          0839  3         THEN
;  721          0840  4             BEGIN                   ! Normal text characters are packed directly
;  722          0841  4             CH$WCHAR_A (.character, line_ptr);
;  723          0842  4             line_cnt = .line_cnt + 1
;  724          0843  4             END
;  725          0844  3         ELSE
;  726          0845  4             BEGIN                   ! Special case flag handling goes on here
;  727          0846  4             character = CH$RCHAR_A (strg_ptr);
;  728          0847  4             i=.i+1;
;  729          0848  4
;  730          0849  4             !Process all of the alternatives
;  731          0850  4             SELECTONE .character OF
;  732          0851  4                 SET
;  733          0852  4
;  734        U 0853  4 %IF DSRPLUS %THEN
;  735        U 0854  4                 [%C'A']:          ! Input .BRN File Specification
;  736        U 0855  4                     BEGIN
;  737        U 0856  4                     BIND
;  738        U 0857  4                         file_spec_descr = brniob [iob$t_resultant] : $STR_DESCRIPTOR ();
;  739        U 0858  4
;  740        U 0859  4                     line_cnt =  .line_cnt +
;  741        U 0860  4                                 pacstr (.file_spec_descr [str$a_pointer]
;  742        U 0861  4                                        ,.file_spec_descr [str$h_length]
;  743        U 0862  4                                        ,line_ptr);
;  744        U 0863  4                     END;
;  745          0864  4 %FI
;  746          0865  4                 [%C'B']:          ! Output .BRN File Specification
;  747          0866  5                     BEGIN
;  748          0867  5                     BIND
;  749          0868  5                         file_spec_descr = brnoob [iob$t_resultant] : $STR_DESCRIPTOR ();
```

```
;  750     0869  5
;  751     0870  5                    line_cnt =  .line_cnt +
;  752     0871  5                                pacstr (.file_spec_descr [str$a_pointer]
;  753     0872  5                                       ,.file_spec_descr [str$h_length]
;  754     0873  5                                       ,line_ptr);
;  755     0874  4                    END;
;  756     0875  4
;  757     0876  4                [%C'C']:        ! Input Sequence Number
;  758     0877  4                    line_cnt = .line_cnt+pacbas (.irac_iseqn, line_ptr, 10);
;  759     0878  4
;  760     0879  4                [%C'F']:        ! Current Input File Name
;  761     0880  4                    line_cnt = .line_cnt + pacstr (.irac_fspecp
;  762     0881  4                                                  ,.irac_fspecc
;  763     0882  4                                                  ,line_ptr);
;  764     0883  4
;  765     0884  4                [%C'O']:        ! Output File Specification
;  766     0885  5                    BEGIN
;  767     0886  5                    BIND
;  768     0887  5                        file_spec_descr = rnoiob [iob$t_resultant] : $STR_DESCRIPTOR ();
;  769     0888  5
;  770     0889  5                    line_cnt =  .line_cnt +
;  771     0890  5                                pacstr (.file_spec_descr [str$a_pointer]
;  772     0891  5                                       ,.file_spec_descr [str$h_length]
;  773     0892  5                                       ,line_ptr);
;  774     0893  4                    END;
;  775     0894  4
;  776     0895  4                [%C'I']:        ! Input Page Number
;  777     0896  4                    line_cnt = .line_cnt + pacbas (.irac_ipagen, line_ptr, 10);
;  778     0897  4
;  779     0898  4                [%C'L']:        ! Output Line Number
;  780     0899  4                    line_cnt = .line_cnt + pacbas (.phan_lines_tp, line_ptr, 10);
;  781     0900  4
;  782     0901  4                [%C'N']:        ! Numeric passed argument
;  783     0902  4                    line_cnt = .line_cnt + pacbas (.arg, line_ptr, 10);
;  784     0903  4
;  785     0904  4                [%C'P']:        ! Current Output Page Number
;  786     0905  4                    line_cnt = .line_cnt + pacpag (pagen, line_ptr);
;  787     0906  4
;  788     0907  4                [%C'S']:        ! String passed argument
;  789     0908  4
;  790     0909  4                    ! In no case insert more than 100 characters.
;  791     0910  4                    !            (Defensive coding.)
;  792     0911  4
;  793     0912  4                    line_cnt = .line_cnt + pacstr (.arg
;  794     0913  4                                                  ,MIN (100, .len)
;  795     0914  4                                                  ,line_ptr);
;  796     0915  4
;  797     0916  4                [%C'V']:        ! Insert version number of program
;  798     0917  5                    BEGIN
;  799     0918  5                    EXTERNAL
;  800     0919  5                        rnovrl,         ! Length of version number string
;  801     0920  5                        rnovrp;         ! CH$PTR to string
;  802     0921  5
;  803     0922  5                    line_cnt = .line_cnt + pacstr (.rnovrp, .rnovrl, line_ptr);
;  804     0923  4                    END;
;  805     0924  4
;  806     0925  4                [%C'X']:        ! Insert an "s" if %N was other then 1.
```

```
;  807     0926  4                    IF .arg NEQ 1
;  808     0927  4                    THEN
;  809     0928  5                        BEGIN
;  810     0929  5                        CH$WCHAR_A (%C's', line_ptr);
;  811     0930  5                        line_cnt=.line_cnt+1
;  812     0931  4                        END;
;  813     0932  4
;  814     0933  4                [OTHERWISE]:    ! Unrecognized flag, treat as text
;  815     0934  5                    BEGIN
;  816     0935  5                    CH$WCHAR_A (%C'%',line_ptr);
;  817     0936  5                    CH$WCHAR_A (.character, line_ptr);
;  818     0937  5                    line_cnt = .line_cnt+2
;  819     0938  4                    END;
;  820     0939  4
;  821     0940  4                TES
;  822     0941  4
;  823     0942  4            END
;  824     0943  4
;  825     0944  2        END;
;  826     0945  2
;  827     0946  2    RETURN .line_cnt;
;  828     0947  1    END;                            !End of SCNMSG


                                                    .EXTRN  RNOVRL, RNOVRP

                                    00FC 00000 SCNMSG: .WORD   Save R2,R3,R4,R5,R6,R7                  ; 0756
                 57 00000000G   EF   9E 00002       MOVAB   IRAC+8, R7                                 ;
                         04     AC   DD 00009       PUSHL   LINE                                       ; 0823
                                52   D4 0000C       CLRL    LINE_CNT                                   ; 0824
                 55         08  AC   D0 0000E       MOVL    MESSAGE, STRG_PTR                          ; 0825
                 56             85   9A 00012       MOVZBL  (STRG_PTR)+, STRG_CNT                      ; 0828
                                54   D4 00015       CLRL    I                                          ; 0831
                            0135 31   00017       BRW     21$                                        ;
                 53             85   9A 0001A 1$:   MOVZBL  (STRG_PTR)+, CHARACTER                    ; 0836
                 25             53   D1 0001D       CMPL    CHARACTER, #37                             ; 0838
                                07   13 00020       BEQL    2$                                         ;
              00 BE             53   90 00022       MOVB    CHARACTER, @LINE_PTR                       ; 0841
                            0111 31   00026       BRW     19$                                        ;
                 53             85   9A 00029 2$:   MOVZBL  (STRG_PTR)+, CHARACTER                    ; 0846
                                54   D6 0002C       INCL    I                                          ; 0847
00000042         8F             53   D1 0002E       CMPL    CHARACTER, #66                             ; 0865
                                11   12 00035       BNEQ    3$                                         ;
                                5E   DD 00037       PUSHL   SP                                         ; 0871
                 7E 00000000G   EF   3C 00039       MOVZWL  FILE_SPEC_DESCR, -(SP)                     ; 0872
                    00000000G   EF   DD 00040       PUSHL   FILE_SPEC_DESCR+4                          ; 0871
                                3C   11 00046       BRB     6$                                         ;
00000043         8F             53   D1 00048 3$:   CMPL    CHARACTER, #67                             ; 0876
                                09   12 0004F       BNEQ    4$                                         ;
                                0A   DD 00051       PUSHL   #10                                        ; 0877
                         04     AE   9F 00053       PUSHAB  LINE_PTR                                   ;
                                67   DD 00056       PUSHL   IRAC+8                                     ;
                                66   11 00058       BRB     10$                                        ;
00000046         8F             53   D1 0005A 4$:   CMPL    CHARACTER, #70                             ; 0879
                                08   12 00061       BNEQ    5$                                         ;
                                5E   DD 00063       PUSHL   SP                                         ; 0880
```

```
              7E        08  A7  7D 00065          MOVQ     IRAC+16, -(SP)             ;
                            19  11 00069          BRB      6$                         ;
        0000004F   8F       53  D1 0006B  5$:     CMPL     CHARACTER, #79             ; 0884
                            12  12 00072          BNEQ     7$                         ;
     50 0000000G   EF       1C  C1 00074          ADDL3    #28, RNOIOB, R0            ; 0887
                            5E  DD 0007C          PUSHL    SP                         ; 0890
              7E            60  3C 0007E          MOVZWL   (R0), -(SP)                ; 0891
                        04  A0  DD 00081          PUSHL    4(R0)                      ; 0890
                            7B  11 00084  6$:     BRB      14$                        ;
        00000049   8F       53  D1 00086  7$:     CMPL     CHARACTER, #73             ; 0895
                            0A  12 0008D          BNEQ     8$                         ;
                            0A  DD 0008F          PUSHL    #10                        ; 0896
                        04  AE  9F 00091          PUSHAB   LINE_PTR                   ;
                        04  A7  DD 00094          PUSHL    IRAC+12                    ;
                            27  11 00097          BRB      10$                        ;
        0000004C   8F       53  D1 00099  8$:     CMPL     CHARACTER, #76             ; 0898
                            0D  12 000A0          BNEQ     9$                         ;
                            0A  DD 000A2          PUSHL    #10                        ; 0899
                        04  AE  9F 000A4          PUSHAB   LINE_PTR                   ;
                 0000000G   EF  DD 000A7          PUSHL    PHAN+12                    ;
                            11  11 000AD          BRB      10$                        ;
        0000004E   8F       53  D1 000AF  9$:     CMPL     CHARACTER, #78             ; 0901
                            11  12 000B6          BNEQ     11$                        ;
                            0A  DD 000B8          PUSHL    #10                        ; 0902
                        04  AE  9F 000BA          PUSHAB   LINE_PTR                   ;
                        0C  AC  DD 000BD          PUSHL    ARG                        ;
        0000000G   EF       03  FB 000C0  10$:    CALLS    #3, PACBAS                 ;
                            58  11 000C7          BRB      17$                        ;
        00000050   8F       53  D1 000C9  11$:    CMPL     CHARACTER, #80             ; 0904
                            11  12 000D0          BNEQ     12$                        ;
                            5E  DD 000D2          PUSHL    SP                         ; 0905
                 0000000G   EF  9F 000D4          PUSHAB   PAGEN                      ;
        0000000G   EF       02  FB 000DA          CALLS    #2, PACPAG                 ;
                            3E  11 000E1          BRB      17$                        ;
        00000053   8F       53  D1 000E3  12$:    CMPL     CHARACTER, #83             ; 0907
                            17  12 000EA          BNEQ     15$                        ;
                            5E  DD 000EC          PUSHL    SP                         ; 0912
                        10  AC  DD 000EE          PUSHL    LEN                        ; 0913
        00000064   8F       6E  D1 000F1          CMPL     (SP), #100                 ;
                            04  15 000F8          BLEQ     13$                        ;
              6E        64  8F  9A 000FA          MOVZBL   #100, (SP)                 ;
                        0C  AC  DD 000FE  13$:    PUSHL    ARG                        ; 0912
                            17  11 00101  14$:    BRB      16$                        ;
        00000056   8F       53  D1 00103  15$:    CMPL     CHARACTER, #86             ; 0916
                            1A  12 0010A          BNEQ     18$                        ;
                            5E  DD 0010C          PUSHL    SP                         ; 0922
                 0000000G   EF  DD 0010E          PUSHL    RNOVRL                     ;
                 0000000G   EF  DD 00114          PUSHL    RNOVRP                     ;
        0000000G   EF       03  FB 0011A  16$:    CALLS    #3, PACSTR                 ;
                   52       50  C0 00121  17$:    ADDL2    R0, LINE_CNT               ;
                            29  11 00124          BRB      21$                        ; 0850
        00000058   8F       53  D1 00126  18$:    CMPL     CHARACTER, #88             ; 0925
                            11  12 0012D          BNEQ     20$                        ;
              01        0C  AC  D1 0012F          CMPL     ARG, #1                    ; 0926
                            1A  13 00133          BEQL     21$                        ;
           00   BE      73  8F  90 00135          MOVB     #115, @LINE_PTR            ; 0929
                            6E  D6 0013A  19$:    INCL     LINE_PTR                   ;
```

```
                                                52  D6 0013C          INCL    LINE_CNT                                    ; 0930
                                                0F  11 0013E          BRB     21$                                         ; 0926
                                 00    BE       25  90 00140 20$:     MOVB    #37, @LINE_PTR                              ; 0935
                                                6E  D6 00144          INCL    LINE_PTR                                    ;
                                 00    BE       53  90 00146          MOVB    CHARACTER, @LINE_PTR                        ; 0936
                                                6E  D6 0014A          INCL    LINE_PTR                                    ;
                                       52       02  C0 0014C          ADDL2   #2, LINE_CNT                                ; 0937
        FEC5                    54     01       56  F1 0014F 21$:     ACBL    STRG_CNT, #1, I, 1$                         ; 0838
                                       50       52  D0 00155          MOVL    LINE_CNT, R0                                ; 0946
                                                    04 00158          RET                                                 ; 0947

; Routine Size:  345 bytes,    Routine Base:  $CODE$ + 0206

;  829          0948  1
```

```
;  831          0949  1 %SBTTL 'REPORT_SECONDARY_MESSAGE -- if present, get and output it'
;  832          0950  1 ROUTINE report_secondary_message (msgcode) =
;  833          0951  1
;  834          0952  1 !++
;  835          0953  1 ! FUNCTIONAL DESCRIPTION:
;  836          0954  1 !
;  837          0955  1 !       If the MSGCODE is nonzero, it identifies an error message.  Capture
;  838          0956  1 !       that error message and report it using a call to ERM.
;  839          0957  1 !
;  840          0958  1 !       This secondary error message handler was implemented to get the
;  841          0959  1 !       detailed reason for issuing the following error messages:
;  842          0960  1 !
;  843          0961  1 !           RNFCOB, RNFCOF, RNFCOI, RNFCOO, RNFCOR, RNFCOT, or RNFCOX.
;  844          0962  1 !
;  845          0963  1 ! FORMAL PARAMETERS:
;  846          0964  1 !
;  847          0965  1 !       MSGCODE - An error message code.  For the time being, we assume it's
;  848          0966  1 !                 within the scope of XPORTs error messages.
;  849          0967  1 !
;  850          0968  1 ! IMPLICIT INPUTS:      None
;  851          0969  1 !
;  852          0970  1 ! IMPLICIT OUTPUTS:     None
;  853          0971  1 !
;  854          0972  1 ! ROUTINE VALUE:
;  855          0973  1 !
;  856          0974  1 !       Returns TRUE if the message was there and reported.
;  857          0975  1 !       Returns FALSE if the code was zero or otherwise not reported.
;  858          0976  1 !
;  859          0977  1 ! SIDE EFFECTS:         None
;  860          0978  1 !--
;  861          0979  1
;  862          0980  2     BEGIN
;  863          0981  2     EXTERNAL LITERAL
;  864          0982  2         rnfstr; ! I - "%S"
;  865          0983  2
;  866          0984  2     ! Initialize message string descriptor:
;  867          0985  2     w_desc1 [str$b_dtype] = str$k_dtype_t;      ! ASCII text (8-bit)
;  868          0986  2     w_desc1 [str$b_class] = str$k_class_f;      ! Fixed (Scalar) String Descriptor
;  869          0987  2     w_desc1 [str$a_pointer] = w_buffer1 [0];    ! First byte of char. buffer
;  870          0988  2     w_desc1 [str$h_length] = 130;               ! Size of buffer
;  871          0989  2
;  872          0990  2     IF .msgcode NEQ 0
;  873          0991  2     THEN        ! If there is at least something in the code,
;  874          0992  3         BEGIN                   ! attempt to capture the message and report it.
;  875          0993  3
;  876          0994  3 %IF %BLISS(BLISS32) %THEN
;  877          0995  3         sys$getmsg  (.msgcode
;  878          0996  3                     ,w_desc1 [str$h_length]
;  879          0997  3                     ,w_desc1
;  880          0998  3                     ,1
;  881          0999  3                     ,0          );
;  882          1000  3
;  883     U    1001  3 %ELSE
;  884     U    1002  3         XPO$XMSG (.msgcode, w_desc1 );          ! Get message from XPORT
;  885          1003  3 %FI
;  886          1004  3
;  887          1005  3         erm (RNFSTR, .w_desc1 [str$a_pointer], .w_desc1 [str$h_length]);
```

```
;  888          1006  3          RETURN TRUE;                            ! Return code indicating that a
;  889          1007  3          END
;  890          1008  2      ELSE
;  891          1009  2          RETURN FALSE;                           ! Return code indicating that a
;  892          1010  2                                                  ! message has NOT been output.
;  893          1011  1      END;                            !End of SCNMSG


                                        0004 00000 REPORT_SECONDARY_MESSAGE:
                                                           .WORD   Save R2                                 ; 0950
                              52 00000000'   EF  9E 00002          MOVAB   W_DESC1, R2
                              62 010E0082    8F  D0 00009          MOVL    #17694850, W_DESC1               ; 0988
                        04    A2     FF7C    C2  9E 00010          MOVAB   W_BUFFER1, W_DESC1+4             ; 0987
                                        04    AC  D5 00016          TSTL    MSGCODE                          ; 0990
                                              26  13 00019          BEQL    1$
                              7E              01  7D 0001B          MOVQ    #1, -(SP)                        ; 0996
                                              52  DD 0001E          PUSHL   R2
                                              52  DD 00020          PUSHL   R2
                                        04    AC  DD 00022          PUSHL   MSGCODE
                  00000000G   EF              05  FB 00025          CALLS   #5, SYS$GETMSG
                              7E              62  3C 0002C          MOVZWL  W_DESC1, -(SP)                   ; 1005
                                        04    A2  DD 0002F          PUSHL   W_DESC1+4
                                   00000000G  8F  DD 00032          PUSHL   #RNFSTR
                         FD42   CF              03  FB 00038          CALLS   #3, ERM
                              50              01  D0 0003D          MOVL    #1, R0                           ; 1009
                                              04  00040          RET
                                              50  D4 00041 1$:     CLRL    R0
                                              04  00043          RET                                      ; 1011

; Routine Size:  68 bytes,    Routine Base:  $CODE$ + 035F

;  894          1012  1
;  895          1013  1 END                                !End of module
;  896          1014  0 ELUDOM



;                                   PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  $OWN$                         648  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                        931  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)



;                                 Library Statistics
;
;                                 -------- Symbols --------      Pages       Processing
```

```
;       File                                    Total    Loaded  Percent      Mapped      Time
;
;  _$255$DUA28:[SYSLIB]XPORT.L32;1                590        89       15         252      00:00.2
;  _$255$DUA28:[RUNOFF.SRC]DSRLIB.L32;1           1248        29        2          86      00:00.3
```

```
;                                       COMMAND QUALIFIERS

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

; Size:          931 code + 648 data bytes
; Run Time:        00:19.9
; Elapsed Time:    00:53.5
; Lines/CPU Min:    3055
; Lexemes/CPU-Min: 28149
; Memory Used:  149 pages
; Compilation Complete
```