```
NNN           NNN   MMM           MMM   LLL
NNN           NNN   MMM           MMM   LLL
NNN           NNN   MMM           MMM   LLL
NNN           NNN   MMMMMM    MMMMMM   LLL
NNN           NNN   MMMMMM    MMMMMM   LLL
NNN           NNN   MMMMMM    MMMMMM   LLL
NNNNNN        NNN   MMM   MMM   MMM   LLL
NNNNNN        NNN   MMM   MMM   MMM   LLL
NNNNNN        NNN   MMM   MMM   MMM   LLL
NNN   NNN     NNN   MMM         MMM   LLL
NNN   NNN     NNN   MMM         MMM   LLL
NNN   NNN     NNN   MMM         MMM   LLL
NNN       NNNNNN   MMM         MMM   LLL
NNN       NNNNNN   MMM         MMM   LLL
NNN       NNNNNN   MMM         MMM   LLL
NNN           NNN   MMM         MMM   LLL
NNN           NNN   MMM         MMM   LLL
NNN           NNN   MMM         MMM   LLL
NNN           NNN   MMM         MMM   LLLLLLLLLLLLLLL
NNN           NNN   MMM         MMM   LLLLLLLLLLLLLLL
NNN           NNN   MMM         MMM   LLLLLLLLLLLLLLL
```

FILEID**NMLBLDMSG


```
NN       NN  MM       MM  LL          BBBBBBBB    LL          DDDDDDDD    MM       MM    SSSSSSSS    GGGGGGGG
NN       NN  MM       MM  LL          BBBBBBBB    LL          DDDDDDDD    MM       MM    SSSSSSSS    GGGGGGGG
NN       NN  MMMM   MMMM  LL          BB      BB  LL          DD      DD  MMMM   MMMM  SS          GG
NN       NN  MMMM   MMMM  LL          BB      BB  LL          DD      DD  MMMM   MMMM  SS          GG
NNNN     NN  MM  MM  MM  LL          BB      BB  LL          DD      DD  MM  MM  MM  SS          GG
NNNN     NN  MM  MM  MM  LL          BB      BB  LL          DD      DD  MM  MM  MM  SS          GG
NN  NN   NN  MM      MM  LL          BBBBBBBB    LL          DD      DD  MM      MM    SSSSSS    GG
NN  NN   NN  MM      MM  LL          BBBBBBBB    LL          DD      DD  MM      MM    SSSSSS    GG
NN    NNNN  MM      MM  LL          BB      BB  LL          DD      DD  MM      MM          SS  GG  GGGGGG
NN    NNNN  MM      MM  LL          BB      BB  LL          DD      DD  MM      MM          SS  GG  GGGGGG
NN      NN  MM      MM  LL          BB      BB  LL          DD      DD  MM      MM          SS  GG      GG      ....
NN      NN  MM      MM  LL          BB      BB  LL          DD      DD  MM      MM          SS  GG      GG      ....
NN      NN  MM      MM  LLLLLLLLLL  BBBBBBBB    LLLLLLLLLL  DDDDDDDD    MM      MM    SSSSSSSS      GGGGGG      ....
NN      NN  MM      MM  LLLLLLLLLL  BBBBBBBB    LLLLLLLLLL  DDDDDDDD    MM      MM    SSSSSSSS      GGGGGG      ....

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 'NML Network message builder module'
;    2          0002  0 MODULE NML$BLDMSG (
;    3          0003  0                 LANGUAGE (BLISS32),
;    4          0004  0                 ADDRESSING_MODE (NONEXTERNAL=GENERAL),
;    5          0005  0                 ADDRESSING_MODE (EXTERNAL=GENERAL),
;    6          0006  0                 IDENT = 'V04-000'
;    7          0007  0                 ) =
;    8          0008  1 BEGIN
;    9          0009  1 !
;   10          0010  1 !*****************************************************************************
;   11          0011  1 !*                                                                           *
;   12          0012  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;   13          0013  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;   14          0014  1 !*  ALL RIGHTS RESERVED.                                                      *
;   15          0015  1 !*                                                                           *
;   16          0016  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
;   17          0017  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;   18          0018  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
;   19          0019  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
;   20          0020  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
;   21          0021  1 !*  TRANSFERRED.                                                             *
;   22          0022  1 !*                                                                           *
;   23          0023  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
;   24          0024  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
;   25          0025  1 !*  CORPORATION.                                                             *
;   26          0026  1 !*                                                                           *
;   27          0027  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
;   28          0028  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
;   29          0029  1 !*                                                                           *
;   30          0030  1 !*                                                                           *
;   31          0031  1 !*****************************************************************************
;   32          0032  1 !
;   33          0033  1
;   34          0034  1 !++
;   35          0035  1 ! FACILITY:  DECnet-VAX Network Management Listener
;   36          0036  1 !
;   37          0037  1 !
;   38          0038  1 ! ABSTRACT:
;   39          0039  1 !
;   40          0040  1 !       This module contains routines to build NICE response messages
;   41          0041  1 !       and miscellaneous routines for debugging.
;   42          0042  1 !
;   43          0043  1 ! ENVIRONMENT:  VAX/VMS Operating System
;   44          0044  1 !
;   45          0045  1 ! AUTHOR:  Distributed Systems Software Engineering
;   46          0046  1 !
;   47          0047  1 ! CREATION DATE:  28-Jan-1980
;   48          0048  1 !
;   49          0049  1 ! MODIFIED BY:
;   50          0050  1 !
;   51          0051  1 !       V03-012 MKP0012         Kathy Perko             2-Jan-1984
;   52          0052  1 !               Convert from old $TRNLOG system service to new $TRNLNM
;   53          0053  1 !               system service for translating logical names.
;   54          0054  1 !
;   55          0055  1 !       V03-011 MKP0011         Kathy Perko             4-Aug-1983
;   56          0056  1 !               Change format of node permanent database to use multiple ISAM
;   57          0057  1 !               keys for better performance.  Also, give NML a message file.
```

```
;    58          0058  1 !
;    59          0059  1 !       V03-010 MKP0010         Kathy Perko             11-April-1982
;    60          0060  1 !               Fix response message length for system messages.
;    61          0061  1 !
;    62          0062  1 !       V03-009 MKP0009         Kathy Perko             9-Nov-1982
;    63          0063  1 !               Add code to log all NICE messages to NML$WATCHER
;    64          0064  1 !               if the logical name is defined.
;    65          0065  1 !               Add code to NML logging so that any information that is printed
;    66          0066  1 !               which is past the end of the buffer is zeros.
;    67          0067  1 !
;    68          0068  1 !       V03-008 MKP0008         Kathy Perko             4-Oct-1982
;    69          0069  1 !               Add X29 file dump to logging stuff.
;    70          0070  1 !
;    71          0071  1 !       V03-007 MKP0007         Kathy Perko             13-July-1982
;    72          0072  1 !               Enhance NML$ADDMSGPRM to handle hex image fields.
;    73          0073  1 !
;    74          0074  1 !       V03-006 MKP0006         Kathy Perko             10-Jan-1982
;    75          0075  1 !               Add a message parameter to NML$DEBUG_QIO so I can label
;    76          0076  1 !               what kind of QIO is getting logged.
;    77          0077  1 !
;    78          0078  1 !       V03-005 MKP0005         Kathy Perko             20-Dec-1981
;    79          0079  1 !               Change debug logging so that it can handle no NFB descriptor
;    80          0080  1 !               address.  This allows NML to dump just the QIO status and
;    81          0081  1 !               IOSB for MOP QIOs.
;    82          0082  1 !
;    83          0083  1 !       V03-004 MKP0004         Kathy Perko             03-Nov-1981
;    84          0084  1 !               Add support for secondary (RMS) lines of message text in a
;    85          0085  1 !               NICE reply message.
;    86          0086  1 !
;    87          0087  1 !       V03-003 MKP0003         Kathy Perko             20-Sept-1991
;    88          0088  1 !               Fix logging so QIO info is dumped.
;    89          0089  1 !
;    90          0090  1 !       V03-002 MKP0002         Kathy Perko             26-July-1981
;    91          0091  1 !               Enhance logging to dump contents of QIO buffers, and
;    92          0092  1 !               dump buffers right to left with ASCII on the right.
;    93          0093  1 !
;    94          0094  1 !       V03-001 MKP0001         Kathy Perko             26-July-1981
;    95          0095  1 !               Fix logging so permanent data base files aren't
;    96          0096  1 !               opened and closed every time.
;    97          0097  1 !--
;    98          0098  1
```

```
;  100     0099  1 %SBTTL 'Declarations'
;  101     0100  1
;  102     0101  1 !
;  103     0102  1 ! TABLE OF CONTENTS:
;  104     0103  1 !
;  105     0104  1
;  106     0105  1 FORWARD ROUTINE
;  107     0106  1     NML$BLD_REPLY,
;  108     0107  1     NML$ADD_MSG_TXT,
;  109     0108  1     NML$ADDMSGPRM,
;  110     0109  1     NML$ADDMSGCOU,
;  111     0110  1     NML$ERROR_1           : NOVALUE,
;  112     0111  1     NML$ERROR_2           : NOVALUE,
;  113     0112  1     NML$DEBUG_TXT         : NOVALUE,
;  114     0113  1     NML$DEBUG_MSG         : NOVALUE,
;  115     0114  1     NML$DEBUG_QIO         : NOVALUE,
;  116     0115  1     NML$DUMP_QIO_BUFS     : NOVALUE,
;  117     0116  1     NML$LOGALLPDB         : NOVALUE,
;  118     0117  1     NML$FILEDMP           : NOVALUE,
;  119     0118  1     NML$LOGFILEOP         : NOVALUE,
;  120     0119  1     NML$LOGRECORDOP       : NOVALUE,
;  121     0120  1     NML$APPENDTXT         : NOVALUE,
;  122     0121  1     NML$TRNLOGNUM;
;  123     0122  1
;  124     0123  1 !
;  125     0124  1 ! INCLUDE FILES:
;  126     0125  1 !
;  127     0126  1
;  128     0127  1 LIBRARY 'LIB$:NMLLIB.L32';
;  129     0128  1 LIBRARY 'SHRLIB$:NMALIBRY.L32';
;  130     0129  1 LIBRARY 'SYS$LIBRARY:STARLET.L32';
;  131     0130  1
;  132     0131  1 !
;  133     0132  1 ! EXTERNAL REFERENCES:
;  134     0133  1 !
;  135     0134  1
;  136     0135  1 $NML_EXTDEF;
;  137     0136  1
;  138     0137  1 EXTERNAL
;  139     0138  1     NML$GQ_PROPRVMSK : BBLOCK [8],      ! Process privilege mask
;  140     0139  1     NML$GW_WATCHER_CHAN: WORD;          ! Channel assigned for NML$WATCHER.
;  141     0140  1
;  142     0141  1 EXTERNAL ROUTINE
;  143     0142  1     LIB$CVT_HTB           : ADDRESSING_MODE (GENERAL),
;  144     0143  1     LIB$PUT_OUTPUT        : ADDRESSING_MODE (GENERAL),
;  145     0144  1     NML$CLOSEFILE,
;  146     0145  1     NMA$OPENFILE,
;  147     0146  1     NMA$READREC;
;  148     0147  1
```

```
;  150          0148  1 %SBTTL 'NML$BLD_REPLY  Build NICE response message'
;  151          0149  1 GLOBAL ROUTINE NML$BLD_REPLY (MSGBLK, MSGLEN) =
;  152          0150  1
;  153          0151  1 !++
;  154          0152  1 ! FUNCTIONAL DESCRIPTION:
;  155          0153  1 !
;  156          0154  1 !       This routine builds a NICE response message based on the
;  157          0155  1 !       message segment block.
;  158          0156  1 !
;  159          0157  1 ! FORMAL PARAMETERS:
;  160          0158  1 !
;  161          0159  1 !       MSGBLK          Address of the message segment block (MSB).
;  162          0160  1 !       MSGLEN          Address of longword to return the total size of
;  163          0161  1 !                       the message that was built.
;  164          0162  1 !
;  165          0163  1 ! IMPLICIT INPUTS:
;  166          0164  1 !
;  167          0165  1 !       NONE
;  168          0166  1 !
;  169          0167  1 ! IMPLICIT OUTPUTS:
;  170          0168  1 !
;  171          0169  1 !       NML$AB_SNDBUFFER contains the NICE reply message built as described in
;  172          0170  1 !       the message segment block.
;  173          0171  1 !
;  174          0172  1 ! ROUTINE VALUE:
;  175          0173  1 ! COMPLETION CODES:
;  176          0174  1 !
;  177          0175  1 !       NONE
;  178          0176  1 !
;  179          0177  1 ! SIDE EFFECTS:
;  180          0178  1 !
;  181          0179  1 !       The NICE response message is in NML$AB_SNDBUFFER.
;  182          0180  1 !
;  183          0181  1 !--
;  184          0182  1
;  185          0183  2 BEGIN
;  186          0184  2
;  187          0185  2 MAP
;  188          0186  2     msgblk : REF BBLOCK;
;  189          0187  2
;  190          0188  2 LOCAL
;  191          0189  2     bufcnt  : SIGNED,                       ! Message length counter
;  192          0190  2     len     : BYTE,                         ! Temporary string length
;  193          0191  2     msg_count_ptr,                          ! Error text length pointer
;  194          0192  2     in_ptr,                                 ! Input text pointer
;  195          0193  2     out_ptr;                                ! Output message pointer
;  196          0194  2 !
;  197          0195  2 ! The MSB longword mask determines the message fields that are
;  198          0196  2 ! described in the following longwords.  The status code is always
;  199          0197  2 ! required.
;  200          0198  2 !
;  201          0199  2 bufcnt = 0;                                 ! Initialize buffer count
;  202          0200  2 out_ptr = nml$ab_sndbuffer;                 ! Get output buffer pointer
;  203          0201  2 CH$WCHAR_A (.msgblk [msb$b_code], out_ptr);  ! Add return code
;  204          0202  2 bufcnt = .bufcnt + 1;                       ! Increment message count
;  205          0203  2 !
;  206          0204  2 ! Check for detail field.
```

```
;  207          0205  2 !
;  208          0206  2 IF .msgblk [msb$v_det_fld] THEN
;  209          0207  3     BEGIN
;  210          0208  3     !
;  211          0209  3     ! Move the detail word into the message buffer.
;  212          0210  3     !
;  213          0211  3     (.out_ptr)<0,16> = .msgblk [msb$w_detail];
;  214          0212  3     out_ptr = .out_ptr + 2;
;  215          0213  3     END
;  216          0214  2 ELSE
;  217          0215  3     BEGIN
;  218          0216  3     !
;  219          0217  3     ! No detail field is specified so add a minus one to the message.
;  220          0218  3     !
;  221          0219  3     (.out_ptr)<0,16> = -1;
;  222          0220  3     out_ptr = .out_ptr + 2;
;  223          0221  2     END;
;  224          0222  2 bufcnt = .bufcnt + 2;                   ! Add detail length to count
;  225          0223  2 !
;  226          0224  2 ! Check for message field if there is room in the buffer.
;  227          0225  2 !
;  228          0226  2 IF .bufcnt LSS nml$k_sndbflen THEN
;  229          0227  3     BEGIN
;  230          0228  3     !
;  231          0229  3     ! Initialize the message length field to zero.  Then save the address
;  232          0230  3     ! of the count so, if there's more than one message to add, the count
;  233          0231  3     ! field can be incremented.
;  234          0232  3     !
;  235          0233  3     msg_count_ptr = .out_ptr;
;  236          0234  3     CH$WCHAR_A (0, out_ptr);
;  237          0235  3     bufcnt = .bufcnt + 1;
;  238          0236  3     IF .msgblk [msb$v_msg_fld] THEN
;  239          0237  3         out_ptr = nml$add_msg_txt
;  240          0238  3                       (.msgblk [msb$l_text],  ! Message code
;  241          0239  3                        bufcnt,                ! Bytes left in NCP response msg
;  242          0240  3                        .msg_count_ptr,        ! Error msg count pointer
;  243          0241  3                        .out_ptr);             ! Message end pointer
;  244          0242  3     !
;  245          0243  3     ! If a secondary status message is requested, then append a CR/LF and the
;  246          0244  3     ! second line of message text to the ASCIC text string in the NICE response.
;  247          0245  3     !
;  248          0246  3     IF .msgblk [msb$v_msg2_fld] THEN            ! If secondary message supplied,
;  249          0247  3         out_ptr = nml$add_msg_txt
;  250          0248  3                       (.msgblk [msb$l_text2], ! Message code
;  251          0249  3                        bufcnt,                ! Bytes left in NCP response msg
;  252          0250  3                        .msg_count_ptr,        ! Error msg count pointer
;  253          0251  3                        .out_ptr);             ! Message end pointer
;  254          0252  2     END;
;  255          0253  2 !
;  256          0254  2 ! If there is room in the buffer check for the entity field.
;  257          0255  2 !
;  258          0256  2 IF .bufcnt LSS nml$k_sndbflen THEN
;  259          0257  3     BEGIN
;  260          0258  3     IF .msgblk [msb$v_entd_fld]
;  261          0259  3     AND (.msgblk [msb$a_entity] NEQA 0) THEN
;  262          0260  4         BEGIN
;  263          0261  4         !
```

```
;  264          0262  4          ! Entity field is ASCID string.
;  265          0263  4          !
;  266          0264  4          BIND
;  267          0265  4              ent_dsc = msgblk [msb$a_entity] : REF DESCRIPTOR;
;  268          0266  4
;  269          0267  4          in_ptr = .ent_dsc [dsc$a_pointer]; ! Get entity pointer
;  270          0268  4          len = .ent_dsc [dsc$w_length];     ! Get length
;  271          0269  4          !
;  272          0270  4          ! If message will not fit in the buffer move the maximum.
;  273          0271  4          !
;  274          0272  4          IF (.bufcnt + .len) GTR nml$k_sndbflen THEN
;  275          0273  4              len = nml$k_sndbflen - .bufcnt;
;  276          0274  4          !
;  277          0275  4          ! Move the count and the entity string into the buffer and add
;  278          0276  4          ! the length to the total.
;  279          0277  4          !
;  280          0278  4          out_ptr = CH$MOVE (.len,
;  281          0279  4                             .in_ptr,
;  282          0280  4                             .out_ptr);
;  283          0281  4          bufcnt = .bufcnt + .len;
;  284          0282  3          END;
;  285          0283  2      END;
;  286          0284  2 !
;  287          0285  2 ! If there is room in the buffer check for the data field.
;  288          0286  2 !
;  289          0287  2 IF .bufcnt LSS nml$k_sndbflen THEN
;  290          0288  3     BEGIN
;  291          0289  3     IF .msgblk [msb$v_data_fld]
;  292          0290  3     AND (.msgblk [msb$a_data] NEQA 0) THEN
;  293          0291  4         BEGIN
;  294          0292  4         !
;  295          0293  4         ! Data field is ASCID string.
;  296          0294  4         !
;  297          0295  4         BIND
;  298          0296  4             datadsc = msgblk [msb$a_data] : REF DESCRIPTOR;
;  299          0297  4
;  300          0298  4         in_ptr = .datadsc [dsc$a_pointer]; ! Get data pointer
;  301          0299  4         len = .datadsc [dsc$w_length];     ! Get length
;  302          0300  4         !
;  303          0301  4         ! If message will not fit in the buffer move the maximum.
;  304          0302  4         !
;  305          0303  4         IF (.bufcnt + .len) LEQ nml$k_sndbflen THEN
;  306          0304  5             BEGIN
;  307          0305  5             !
;  308          0306  5             ! Move the data string into the buffer and add length to total.
;  309          0307  5             !
;  310          0308  5             out_ptr = ch$move (.len,
;  311          0309  5                                .in_ptr,
;  312          0310  5                                .out_ptr);
;  313          0311  5             bufcnt = .bufcnt + .len;
;  314          0312  4             END;
;  315          0313  3         END;
;  316          0314  2     END;
;  317          0315  2 .msglen = .bufcnt;                          ! Return total message size
;  318          0316  2 RETURN nml$_sts_suc                         ! Return success
;  319          0317  1 END;                                ! End of NML$BLD_REPLY
```

```
                                        .TITLE  NML$BLDMSG NML Network message builder module
                                        .IDENT  \V04-000\

                                        .EXTRN  NML$GB_EVTSRCTYP
                                        .EXTRN  NML$GQ_EVTSRCDSC
                                        .EXTRN  NML$GW_EVTCLASS
                                        .EXTRN  NML$GB_EVTMSKTYP
                                        .EXTRN  NML$GQ_EVTMSKDSC
                                        .EXTRN  NML$GW_EVTSNKADR
                                        .EXTRN  NML$GW_ACP_CHAN
                                        .EXTRN  NML$GL_LOGMASK, NML$GQ_ENTSTRDSC
                                        .EXTRN  NML$AB_QIOBUFFER
                                        .EXTRN  NML$GQ_QIOBFDSC
                                        .EXTRN  NML$AB_EXEBUFFER
                                        .EXTRN  NML$GL_EXEDATPTR
                                        .EXTRN  NML$GQ_EXEDATDSC
                                        .EXTRN  NML$GQ_EXEBFDSC
                                        .EXTRN  NML$AB_RCVBUFFER
                                        .EXTRN  NML$GQ_RCVBFDSC
                                        .EXTRN  NML$AB_SNDBUFFER
                                        .EXTRN  NML$GQ_SNDBFDSC
                                        .EXTRN  NML$GL_RCVDATLEN
                                        .EXTRN  NML$AB_CPTABLE, NML$AB_MSGBLOCK
                                        .EXTRN  NML$AB_ENTITY_ID
                                        .EXTRN  NML$AB_QUALIFIER_ID
                                        .EXTRN  NML$AB_ENTITYDATA
                                        .EXTRN  NML$AB_NML_NMV, NML$AB_PRMSEM
                                        .EXTRN  NML$AB_RECBUF, NML$AL_ENTINFTAB
                                        .EXTRN  NML$AL_PERMINFTAB
                                        .EXTRN  NML$AW_PRM_DES, NML$GB_CMD_VER
                                        .EXTRN  NML$GB_ENTITY_CODE
                                        .EXTRN  NML$GB_ENTITY_FORMAT
                                        .EXTRN  NML$GL_QUALIFIER_PST
                                        .EXTRN  NML$GB_QUALIFIER_FORMAT
                                        .EXTRN  NML$GB_FUNCTION
                                        .EXTRN  NML$GB_INFO, NML$GB_OPTIONS
                                        .EXTRN  NML$GL_PRMCODE, NML$GL_PRS_FLGS
                                        .EXTRN  NML$GL_NML_ENTITY
                                        .EXTRN  NML$GQ_NETNAMDSC
                                        .EXTRN  NML$GQ_RECBFDSC
                                        .EXTRN  NML$GW_PRMDESCNT
                                        .EXTRN  NML$GQ_PROPRVMSK
                                        .EXTRN  NML$GW_WATCHER_CHAN
                                        .EXTRN  LIB$CVT_HTB, LIB$PUT_OUTPUT
                                        .EXTRN  NML$CLOSEFILE, NMA$OPENFILE
                                        .EXTRN  NMA$READREC

                                        .PSECT  $CODE$,NOWRT,2

                          03FC 00000    .ENTRY  NML$BLD_REPLY, Save R2,R3,R4,R5,R6,R7,R8,R9 ; 0149
59 00000000V   00   9E 00002    MOVAB   NML$ADD_MSG_TXT, R9                                 ;
               7E   D4 00009    CLRL    BUFCNT                                              ; 0199
50 00000000G   00   9E 0000B    MOVAB   NML$AB_SNDBUFFER, OUT_PTR                           ; 0200
56         04   AC   D0 00012    MOVL    MSGBLK, R6                                          ; 0201
80         04   A6   90 00016    MOVB    4(R6), (OUT_PTR)+                                   ;
               6E   D6 0001A    INCL    BUFCNT                                              ; 0202
```

```
                06              66              01  E1  0001C          BBC     #1, (R6), 1$                            ; 0206
                                60          08  A6  B0  00020          MOVW    8(R6), (OUT_PTR)                        ; 0211
                                                03  11  00024          BRB     2$                                      ; 0206
                                6C                  01  AE  00026 1$:   MNEGW   #1, (OUT_PTR)                       ; 0219
                                50                  02  C0  00029 2$:   ADDL2   #2, OUT_PTR                          ; 0212
                                6E                  02  C0  0002C      ADDL2   #2, BUFCNT                          ; 0222
                00000200        8F                  6E  D1  0002F      CMPL    BUFCNT, #512                        ; 0226
                                                    29  18  00036      BGEQ    4$                                  ;
                                52                  50  D0  00038      MOVL    OUT_PTR, MSG_COUNT_PTR              ; 0233
                                                    80  94  0003B      CLRB    (OUT_PTR)+                          ; 0234
                                                    6E  D6  0003D      INCL    BUFCNT                              ; 0235
                0D              66                  02  E1  0003F      BBC     #2, (R6), 3$                        ; 0236
                                                    50  DD  00043      PUSHL   OUT_PTR                             ; 0241
                                                    52  DD  00045      PUSHL   MSG_COUNT_PTR                       ; 0240
                                            08      AE  9F  00047      PUSHAB  BUFCNT                              ; 0238
                                            0C      A6  DD  0004A      PUSHL   12(R6)                              ;
                                69                      04  FB  0004D  CALLS   #4, NML$ADD_MSG_TXT                 ;
                0D              66                  03  E1  00050 3$:   BBC     #3, (R6), 4$                        ; 0246
                                                    50  DD  00054      PUSHL   OUT_PTR                             ; 0251
                                                    52  DD  00056      PUSHL   MSG_COUNT_PTR                       ; 0250
                                            08      AE  9F  00058      PUSHAB  BUFCNT                              ; 0248
                                            10      A6  DD  0005B      PUSHL   16(R6)                              ;
                                69                      04  FB  0005E  CALLS   #4, NML$ADD_MSG_TXT                 ;
                00000200        8F                  6E  D1  00061 4$:   CMPL    BUFCNT, #512                        ; 0256
                                                    37  18  00068      BGEQ    6$                                  ;
                33              66                  04  E1  0006A      BBC     #4, (R6), 6$                        ; 0258
                                            14      A6  D5  0006E      TSTL    20(R6)                              ; 0259
                                                    2E  13  00071      BEQL    6$                                  ;
                                51          14      A6  D0  00073      MOVL    20(R6), R1                          ; 0267
                                58          04      A1  D0  00077      MOVL    4(R1), IN_PTR                       ;
                                57                  61  90  0007B      MOVB    (R1), LEN                           ; 0268
                                51                  57  9A  0007E      MOVZBL  LEN, R1                             ; 0272
                                51                  6E  C0  00081      ADDL2   BUFCNT, R1                          ;
                00000200        8F                  51  D1  00084      CMPL    R1, #512                            ;
                                                    04  15  0008B      BLEQ    5$                                  ;
                57              00                  6E  83  0008D      SUBB3   BUFCNT, #0, LEN                     ; 0273
                                51                  57  9A  00091 5$:   MOVZBL  LEN, R1                            ; 0278
                60              68                  51  28  00094      MOVC3   R1, (IN_PTR), (OUT_PTR)             ; 0280
                                50                  53  D0  00098      MOVL    R3, OUT_PTR                         ;
                                51                  57  9A  0009B      MOVZBL  LEN, R1                             ; 0281
                                6E                  51  C0  0009E      ADDL2   R1, BUFCNT                          ;
                00000200        8F                  6E  D1  000A1 6$:   CMPL    BUFCNT, #512                        ; 0287
                                                    33  18  000A8      BGEQ    7$                                  ;
                2F              66                  05  E1  000AA      BBC     #5, (R6), 7$                        ; 0289
                                            18      A6  D5  000AE      TSTL    24(R6)                              ; 0290
                                                    2A  13  000B1      BEQL    7$                                  ;
                                51          18      A6  D0  000B3      MOVL    24(R6), R1                          ; 0298
                                58          04      A1  D0  000B7      MOVL    4(R1), IN_PTR                       ;
                                57                  61  90  000BB      MOVB    (R1), LEN                           ; 0299
                                51                  57  9A  000BE      MOVZBL  LEN, R1                             ; 0303
                                51                  6E  C0  000C1      ADDL2   BUFCNT, R1                          ;
                00000200        8F                  51  D1  000C4      CMPL    R1, #512                            ;
                                                    10  14  000CB      BGTR    7$                                  ;
                                51                  57  9A  000CD      MOVZBL  LEN, R1                             ; 0308
                60              68                  51  28  000D0      MOVC3   R1, (IN_PTR), (OUT_PTR)             ; 0310
                                50                  53  D0  000D4      MOVL    R3, OUT_PTR                         ;
                                50                  57  9A  000D7      MOVZBL  LEN, R0                             ; 0311
```

```
                         6E            50  C0 000DA          ADDL2   R0, BUFCNT                              ;
                   08    BC            6E  D0 000DD 7$:      MOVL    BUFCNT, @MSGLEN                         ; 0315
                         50            01  D0 000E1          MOVL    #1, R0                                  ; 0316
                                           04 000E4          RET                                             ; 0317

; Routine Size:  229 bytes,    Routine Base:  $CODE$ + 0000
```

```
;   321          0318  1 %SBTTL 'NML$ADD_MSG_TXT         Add message text to NCP response'
;   322          0319  1 GLOBAL ROUTINE NML$ADD_MSG_TXT (CODE, NICE_BYTES,
;   323          0320  1                                 MSG_COUNT_PTR, MSG_PTR) =
;   324          0321  1
;   325          0322  1 !++
;   326          0323  1 ! FUNCTIONAL DESCRIPTION:
;   327          0324  1 !       This routine performs a $GETMSG system service to retrieve the
;   328          0325  1 !       NML or system message text for the specified status code.
;   329          0326  1 !
;   330          0327  1 ! FORMAL PARAMETERS:
;   331          0328  1 !
;   332          0329  1 !       CODE            Message error code.
;   333          0330  1 !       NICE_BYTES      Address of bytes in NICE response message so far.
;   334          0331  1 !       MSG_COUNT_PTR   Pointer to message count field (if there's
;   335          0332  1 !                       more than one message, the count field must include
;   336          0333  1 !                       all messages.)
;   337          0334  1 !       MSG_POINTER     Pointer to end of message.
;   338          0335  1 !
;   339          0336  1 ! IMPLICIT OUTPUTS:
;   340          0337  1 !       The message text is added to Error Message field of the NCP
;   341          0338  1 !       response message.
;   342          0339  1 !
;   343          0340  1 !--
;   344          0341  1
;   345          0342  2 BEGIN
;   346          0343  2
;   347          0344  2 OWN
;   348          0345  2     msgbuf : BBLOCK [255];
;   349          0346  2
;   350          0347  2 LOCAL
;   351          0348  2     bufdsc : VECTOR [2],                ! System message buffer descriptor
;   352          0349  2     reslen : WORD,                      ! Length of text
;   353          0350  2     crlf;
;   354          0351  2
;   355          0352  2 bufdsc [0] = 255;                       ! Initialize buffer descriptor
;   356          0353  2 bufdsc [1] = msgbuf;
;   357          0354  2 !
;   358          0355  2 ! Retrieve the NML or system message text for the specified error code.
;   359          0356  2 !
;   360        P 0357  2 $GETMSG (MSGID  = .code,
;   361        P 0358  2          MSGLEN = reslen,
;   362          0359  2          BUFADR = bufdsc);
;   363          0360  2 !
;   364          0361  2 ! If message will not fit in the buffer move the maximum.  Just in case
;   365          0362  2 ! there is a <CR><LF>, leave room for it.
;   366          0363  2 !
;   367          0364  2 IF (..nice_bytes + .reslen + 2) GTR nml$k_sndbflen THEN
;   368          0365  2     reslen = nml$k_sndbflen - ..nice_bytes - 2;
;   369          0366  2 !
;   370          0367  2 ! If there's already some error text in the message, add a <CR><LF> before
;   371          0368  2 ! adding the second error message.
;   372          0369  2 !
;   373          0370  2 crlf = 0;
;   374          0371  2 IF .msg_count_ptr NEQ (.msg_ptr - 1) THEN
;   375          0372  3     BEGIN
;   376          0373  3     msg_ptr = CH$MOVE (2, UPLIT BYTE(13, 10), .msg_ptr);
;   377          0374  3     crlf = 2;
```

```
;  378        0375  2       END;
;  379        0376  2  !
;  380        0377  2  ! Move the error message text into the NCP response message.
;  381        0378  2  !
;  382        0379  2  msg_ptr = CH$MOVE (.reslen, msgbuf, .msg_ptr);
;  383        0380  2  .nice_bytes = ..nice_bytes + .reslen + .crlf;    ! Increment buffer space used
;  384        0381  2  (.msg_count_ptr)<0,8> = .(.msg_count_ptr)<0,8>  ! Increment ASCIC string length
;  385        0382  2                               + .reslen +.crlf;
;  386        0383  2  RETURN .msg_ptr;
;  387        0384  1  END;                               ! End of NML$ADD_MSG_TXT


                                                            .PSECT  $PLIT$,NOWRT,NOEXE,2

                                   0A  0D  00000 P.AAA:     .BYTE   13, 10                                  ;

                                                            .PSECT  $OWN$,NOEXE,2

                                           00000 MSGBUF:    .BLKB   255

                                                            .EXTRN  SYS$GETMSG

                                                            .PSECT  $CODE$,NOWRT,2

                                      00FC 00000            .ENTRY  NML$ADD_MSG_TXT, Save R2,R3,R4,R5,R6,R7  ; 0319
                 57  00000000'    00   9E 00002             MOVAB   MSGBUF, R7                               ;
                 5E               0C   C2 00009             SUBL2   #12, SP                                  ;
           04    AE          FF   8F   9A 0000C             MOVZBL  #255, BUFDSC                             ; 0352
           08    AE               67   9E 00011             MOVAB   MSGBUF, BUFDSC+4                         ; 0353
                 7E               0F   7D 00015             MOVQ    #15, -(SP)                               ; 0359
                             0C   AE   9F 00018             PUSHAB  BUFDSC                                   ;
                             0C   AE   9F 0001B             PUSHAB  RESLEN                                   ;
                             04   AC   DD 0001E             PUSHL   CODE                                     ;
        00000000G   00            05   FB 00021             CALLS   #5, SYS$GETMSG                           ;
                    50            6E   3C 00028             MOVZWL  RESLEN, R0                               ; 0364
                    50       08   BC   C0 0002B             ADDL2   @NICE_BYTES, R0                          ;
                    50            02   C0 0002F             ADDL2   #2, R0                                   ;
        00000200    8F            50   D1 00032             CMPL    R0, #512                                 ;
                                  07   15 00039             BLEQ    1$                                       ;
     6E      01FE   8F       08   BC   A3 0003B             SUBW3   @NICE_BYTES, #510, RESLEN                ; 0365
                                  56   D4 00042 1$:         CLRL    CRLF                                     ; 0370
     50        10   AC            01   C3 00044             SUBL3   #1, MSG_PTR, R0                          ; 0371
                    50       0C   AC   D1 00049             CMPL    MSG_COUNT_PTR, R0                        ;
                                  0F   13 0004D             BEQL    2$                                       ;
           10    BC  00000000'    00   B0 0004F             MOVW    P.AAA, @MSG_PTR                          ; 0373
           10    AC               02   C0 00057             ADDL2   #2, MSG_PTR                              ;
                 56               02   D0 0005B             MOVL    #2, CRLF                                 ; 0374
  10     BC      67               6E   28 0005E 2$:         MOVC3   RESLEN, MSGBUF, @MSG_PTR                 ; 0379
           10    AC               53   D0 00063             MOVL    R3, MSG_PTR                              ;
                 50               6E   3C 00067             MOVZWL  RESLEN, R0                               ; 0380
                 50          08   BC   C0 0006A             ADDL2   @NICE_BYTES, R0                          ;
  08     BC      50               56   C1 0006E             ADDL3   CRLF, R0, @NICE_BYTES                    ;
                 50          0C   BC   9A 00073             MOVZBL  @MSG_COUNT_PTR, R0                       ; 0382
                 51               6E   3C 00077             MOVZWL  RESLEN, R1                               ;
                 50               51   C0 0007A             ADDL2   R1, R0                                   ;
  0C     BC      50               56   81 0007D             ADDB3   CRLF, R0, @MSG_COUNT_PTR                 ;
```

```
                                50       10    AC  D0 00082          MOVL    MSG_PTR, R0                     ; 0383
                                                   04 00086          RET                                     ; 0384

; Routine Size:  135 bytes,    Routine Base:  $CODE$ + 00E5
```

```
;  389          0385  1 %SBTTL 'NML$ADDMSGPRM  Add data to NICE response message'
;  390          0386  1 GLOBAL ROUTINE NML$ADDMSGPRM (BUFDSC, MSGSIZE, DATAID, DATATYPE, FLDSIZE, FLDADR) =
;  391          0387  1
;  392          0388  1 !++
;  393          0389  1 ! FUNCTIONAL DESCRIPTION:
;  394          0390  1 !
;  395          0391  1 !       This routine adds a NICE output parameter to the response message.
;  396          0392  1 !       The data id, data type, and the value are added.
;  397          0393  1 !
;  398          0394  1 ! FORMAL PARAMETERS:
;  399          0395  1 !
;  400          0396  1 !       BUFDSC          Address of message buffer descriptor.
;  401          0397  1 !       MSGSIZE         Address of longword containing current (and
;  402          0398  1 !                       resulting message size.
;  403          0399  1 !       DATAID          Parameter code (word value).
;  404          0400  1 !       DATATYPE        Automatic parsing data type (byte value).
;  405          0401  1 !       FLDSIZE         Length of data.
;  406          0402  1 !       FLDADR          Address of data string.
;  407          0403  1 !
;  408          0404  1 ! IMPLICIT INPUTS:
;  409          0405  1 !
;  410          0406  1 !       NONE
;  411          0407  1 !
;  412          0408  1 ! IMPLICIT OUTPUTS:
;  413          0409  1 !
;  414          0410  1 !       Parameter and descriptive information is in buffer described by
;  415          0411  1 !       BUFDSC and the longword count pointed to by MSGSIZE is incremented
;  416          0412  1 !       by the number of bytes moved.
;  417          0413  1 !
;  418          0414  1 ! ROUTINE VALUE:
;  419          0415  1 ! COMPLETION CODES:
;  420          0416  1 !
;  421          0417  1 !       If the parameter will not fit in the buffer then an error
;  422          0418  1 !       (NML$_STS_SIZ) is returned.  Otherwise, success (NML$_STS_SUC)
;  423          0419  1 !       is returned.
;  424          0420  1 !
;  425          0421  1 ! SIDE EFFECTS:
;  426          0422  1 !
;  427          0423  1 !       NONE
;  428          0424  1 !
;  429          0425  1 !--
;  430          0426  1
;  431          0427  2     BEGIN
;  432          0428  2
;  433          0429  2     MAP
;  434          0430  2         BUFDSC    : REF DESCRIPTOR,
;  435          0431  2         DATAID    : WORD,
;  436          0432  2         DATATYPE  : BBLOCK [1];
;  437          0433  2
;  438          0434  2     LOCAL
;  439          0435  2         CNT,                        ! Total parameter data byte count
;  440          0436  2         PTR;                        ! Output message pointer
;  441          0437  2 !
;  442          0438  2 ! Data must fit in buffer.
;  443          0439  2 !
;  444          0440  2     CNT = .FLDSIZE + 3;
;  445          0441  2
```

```
;  446          0442  3     IF .CNT GTRU (.BUFDSC [DSC$W_LENGTH] - ..MSGSIZE)
;  447          0443  2     THEN
;  448          0444  2         RETURN NML$_STS_SIZ;
;  449          0445  2 !
;  450          0446  2 ! Move parameter code (id) and data type into buffer.
;  451          0447  2 !
;  452          0448  2     PTR = .BUFDSC [DSC$A_POINTER] + ..MSGSIZE;
;  453          0449  2     (.PTR)<0,16> = .DATAID;
;  454          0450  2     PTR = .PTR + 2;
;  455          0451  2     CH$WCHAR_A (.DATATYPE [0,0,8,0], PTR);
;  456          0452  2 !
;  457          0453  2 ! If parameter value is an image field (ASCII or binary) then move the count
;  458          0454  2 ! into the buffer.
;  459          0455  2 !
;  460          0456  2     IF (NOT .DATATYPE [NMA$V_PTY_COD]) AND       ! Not coded value and
;  461          0457  3        (.DATATYPE [NMA$V_PTY_ASC] OR             ! Ascii or
;  462          0458  3         .DATATYPE [NMA$V_PTY_TYP] EQL NMA$C_PTY_HI) ! hex image
;  463          0459  2     THEN
;  464          0460  3         BEGIN
;  465          0461  3         CH$WCHAR_A (.FLDSIZE, PTR);
;  466          0462  3         CNT = .CNT + 1;
;  467          0463  2         END;
;  468          0464  2 !
;  469          0465  2 ! Move the parameter value into the buffer.
;  470          0466  2 !
;  471          0467  2     CH$MOVE (.FLDSIZE, .FLDADR, .PTR);
;  472          0468  2     .MSGSIZE = ..MSGSIZE + .CNT;
;  473          0469  2
;  474          0470  2     RETURN NML$_STS_SUC
;  475          0471  2
;  476          0472  1     END;                                  ! End of NML$ADDMSGPRM
```

```
                                        007C 00000          .ENTRY  NML$ADDMSGPRM, Save R2,R3,R4,R5,R6      ; 0386
                    56      14  AC          03  C1 00002    ADDL3   #3, FLDSIZE, CNT                        ; 0440
                        50      04  AC      D0 00007        MOVL    BUFDSC, R0                              ; 0442
                        51          60      3C 0000B        MOVZWL  (R0), R1                                ;
                        51      08  BC      C2 0000E        SUBL2   @MSGSIZE, R1                            ;
                        51          56      D1 00012        CMPL    CNT, R1                                 ;
                                    04      1B 00015        BLEQU   1$                                      ;
                        50          08      CE 00017        MNEGL   #8, R0                                  ; 0444
                                    04      0001A           RET                                             ;
                    50      04  A0      08  BC C1 0001B 1$: ADDL3   @MSGSIZE, 4(R0), PTR                   ; 0448
                        80      0C  AC      B0 00021        MOVW    DATAID, (PTR)+                          ; 0449
                        80      10  AC      90 00025        MOVB    DATATYPE, (PTR)+                        ; 0451
                                    13      19 00029        BLSS    3$                                      ; 0456
                    08      10  AC          06 E0 0002B     BBS     #6, DATATYPE, 2$                        ; 0457
            20      10  AC          0F      00 ED 00030     CMPZV   #0, #15, DATATYPE, #32                  ; 0458
                                    06      12 00036        BNEQ    3$                                      ;
                        80      14  AC      90 00038 2$:    MOVB    FLDSIZE, (PTR)+                         ; 0461
                                    56      D6 0003C        INCL    CNT                                     ; 0462
                    60      18  BC      14  AC 28 0003E 3$: MOVC3   FLDSIZE, @FLDADR, (PTR)                ; 0467
                            08  BC          56 C0 00044     ADDL2   CNT, @MSGSIZE                           ; 0468
                        50          01      D0 00048        MOVL    #1, R0                                  ; 0470
```

```
                                        04 0004B        RET                                     ; 0472

; Routine Size:  76 bytes,    Routine Base:  $CODE$ + 016C
```

```
;  478          0473  1 %SBTTL 'NML$ADDMSGCOU  Add counter data to NICE response message'
;  479          0474  1 GLOBAL ROUTINE NML$ADDMSGCOU (BUFDSC, MSGSIZE, FLDSIZE, FLDADR) =
;  480          0475  1
;  481          0476  1 !++
;  482          0477  1 ! FUNCTIONAL DESCRIPTION:
;  483          0478  1 !
;  484          0479  1 !       This routine moves the counter parameters into the NICE response
;  485          0480  1 !       message.  Since NETACP returns the counters already formatted for
;  486          0481  1 !       the NICE message, it is just a matter of moving the string in
;  487          0482  1 !       without a parameter type or length.
;  488          0483  1 !
;  489          0484  1 ! FORMAL PARAMETERS:
;  490          0485  1 !
;  491          0486  1 !       BUFDSC          Address of message buffer descriptor.
;  492          0487  1 !       MSGSIZE         Address of longword containing current (and resulting)
;  493          0488  1 !                       message length.
;  494          0489  1 !       FLDSIZE         Byte count of counters string.
;  495          0490  1 !       FLDADR          Address of counters string.
;  496          0491  1 !
;  497          0492  1 ! ROUTINE VALUE:
;  498          0493  1 ! COMPLETION CODES:
;  499          0494  1 !
;  500          0495  1 !       If the counters will not fit in the output message buffer then an
;  501          0496  1 !       error (NML$_STS_SIZ) is returned.  Otherwise, success (NML$_STS_SUC)
;  502          0497  1 !       is returned.
;  503          0498  1 !
;  504          0499  1 !--
;  505          0500  1
;  506          0501  2 BEGIN
;  507          0502  2
;  508          0503  2 MAP
;  509          0504  2         BUFDSC : REF DESCRIPTOR;
;  510          0505  2
;  511          0506  2 LOCAL
;  512          0507  2         CNT,                    ! Total counter data length
;  513          0508  2         PTR;                    ! Output message pointer
;  514          0509  2 !
;  515          0510  2 ! Counters string must fit in the buffer.
;  516          0511  2 !
;  517          0512  2 CNT = .FLDSIZE;
;  518          0513  2
;  519          0514  3 IF (.CNT + 2) GTRU (.BUFDSC [DSC$W_LENGTH] - ..MSGSIZE)
;  520          0515  2 THEN
;  521          0516  2         RETURN NML$_STS_SIZ;
;  522          0517  2 !
;  523          0518  2 ! Move the counters string.
;  524          0519  2 !
;  525          0520  2 PTR = .BUFDSC [DSC$A_POINTER] + ..MSGSIZE;
;  526          0521  2 CH$MOVE (.FLDSIZE, .FLDADR, .PTR);
;  527          0522  2 .MSGSIZE = ..MSGSIZE + .CNT;
;  528          0523  2
;  529          0524  2 RETURN NML$_STS_SUC
;  530          0525  2
;  531          0526  1 END;                            ! End of NML$ADDMSGCOU
```

```
                                                 007C 00000          .ENTRY  NML$ADDMSGCOU, Save R2,R3,R4,R5,R6          ; 0474
                                  56      0C     AC   D0 00002          MOVL    FLDSIZE, CNT                               ; 0512
                                  52      02     A6   9E 00006          MOVAB   2(R6), R2                                  ; 0514
                                  51      04     AC   D0 0000A          MOVL    BUFDSC, R1                                 ;
                                  50             61   3C 0000E          MOVZWL  (R1), R0                                   ;
                                  50      08     BC   C2 00011          SUBL2   @MSGSIZE, R0                               ;
                                  50             52   D1 00015          CMPL    R2, R0                                     ;
                                                 04   1B 00018          BLEQU   1$                                         ;
                                  50             08   CE 0001A          MNEGL   #8, R0                                     ; 0516
                                                 04   0001D          RET                                                ;
                          50      04     A1      08   BC   C1 0001E 1$:  ADDL3   @MSGSIZE, 4(R1), PTR                     ; 0520
                          60      10     BC      0C   AC   28 00024     MOVC3   FLDSIZE, @FLDADR, (PTR)                    ; 0521
                                  08     BC             56   C0 0002A     ADDL2   CNT, @MSGSIZE                              ; 0522
                                  50             01   D0 0002E          MOVL    #1, R0                                     ; 0524
                                                 04   00031          RET                                                ; 0526

; Routine Size:  50 bytes,    Routine Base:  $CODE$ + 01B8
```

```
;  533          0527  1 %SBTTL 'NML$ERROR_1  Signal a single byte status message'
;  534          0528  1 GLOBAL ROUTINE NML$ERROR_1 (ERR) : NOVALUE =
;  535          0529  1
;  536          0530  1 !++
;  537          0531  1 ! FUNCTIONAL DESCRIPTION:
;  538          0532  1 !
;  539          0533  1 !       This routine moves an error or status code into the output buffer
;  540          0534  1 !       and sends the message with length of one byte.
;  541          0535  1 !
;  542          0536  1 ! FORMAL PARAMETERS:
;  543          0537  1 !
;  544          0538  1 !       ERR             NICE status code to be transmitted (NMA$C_STS_xxx).
;  545          0539  1 !
;  546          0540  1 ! IMPLICIT INPUTS:
;  547          0541  1 !
;  548          0542  1 !       NONE
;  549          0543  1 !
;  550          0544  1 ! IMPLICIT OUTPUTS:
;  551          0545  1 !
;  552          0546  1 !       NONE
;  553          0547  1 !-
;  554          0548  1 ! ROUTINE VALUE:
;  555          0549  1 ! COMPLETION CODES:
;  556          0550  1 !
;  557          0551  1 !       NONE
;  558          0552  1 !
;  559          0553  1 ! SIDE EFFECTS:
;  560          0554  1 !
;  561          0555  1 !       An error message is signalled to be send by the condition handler.
;  562          0556  1 !
;  563          0557  1 !--
;  564          0558  1
;  565          0559  2     BEGIN
;  566          0560  2
;  567          0561  2     (NML$AB_SNDBUFFER)<0,8> = .ERR;     ! Move status code into buffer
;  568          0562  2
;  569          0563  2     $SIGNAL_MSG (NML$AB_SNDBUFFER, 1);  ! Signal status message
;  570          0564  2
;  571          0565  1     END;                                ! End of NML$ERROR_1


                                        0004 00000      .ENTRY  NML$ERROR_1, Save R2                                    ; 0528
                              52 00000000G  00   9E 00002      MOVAB   NML$AB_SNDBUFFER, R2                                    ;
                              62        04  AC   90 00009      MOVB    ERR, NML$AB_SNDBUFFER                                   ; 0561
                                            01   DD 0000D      PUSHL   #1                                                      ; 0563
                                            52   DD 0000F      PUSHL   R2                                                      ;
                                 01F90000   8F   DD 00011      PUSHL   #33095680                                               ;
                              00000000G  00      03   FB 00017      CALLS   #3, LIB$SIGNAL                                  ;
                                                 04 0001E      RET                                                             ; 0565

; Routine Size:  31 bytes,    Routine Base:  $CODE$ + 01EA
```

```
;   573          0566  1 %SBTTL 'NML$ERROR_2  Signal an error message with detail field'
;   574          0567  1 GLOBAL ROUTINE NML$ERROR_2 (ERR, DET) : NOVALUE =
;   575          0568  1
;   576          0569  1 !++
;   577          0570  1 ! FUNCTIONAL DESCRIPTION:
;   578          0571  1 !
;   579          0572  1 !       This routine moves an error or status code into the output buffer
;   580          0573  1 !       followed by the detail word.
;   581          0574  1 !
;   582          0575  1 ! FORMAL PARAMETERS:
;   583          0576  1 !
;   584          0577  1 !       ERR             NICE status code to be transmitted (NMA$C_STS_xxx).
;   585          0578  1 !       DET             NICE error detail code.
;   586          0579  1 !
;   587          0580  1 ! IMPLICIT INPUTS:
;   588          0581  1 !
;   589          0582  1 !       NONE
;   590          0583  1 !
;   591          0584  1 ! IMPLICIT OUTPUTS:
;   592          0585  1 !
;   593          0586  1 !       NONE
;   594          0587  1 !
;   595          0588  1 ! ROUTINE VALUE:
;   596          0589  1 ! COMPLETION CODES:
;   597          0590  1 !
;   598          0591  1 !       NONE
;   599          0592  1 !
;   600          0593  1 ! SIDE EFFECTS:
;   601          0594  1 !
;   602          0595  1 !       An error message is signalled to be sent by the condition handler.
;   603          0596  1 !
;   604          0597  1 !--
;   605          0598  1
;   606          0599  2     BEGIN
;   607          0600  2 !
;   608          0601  2 ! Move the error code and the detail code into the buffer.
;   609          0602  2 !
;   610          0603  2     (NML$AB_SNDBUFFER)<0,8> = .ERR;
;   611          0604  2     (NML$AB_SNDBUFFER + 1)<0,16> = .DET;
;   612          0605  2 !
;   613          0606  2 ! Signal the message.
;   614          0607  2 !
;   615          0608  2     $SIGNAL_MSG (NML$AB_SNDBUFFER, 3);
;   616          0609  2
;   617          0610  1     END;                                   ! End of NML$ERROR_2


                                    0004 00000          .ENTRY  NML$ERROR_2, Save R2                                    ; 0567
                          52 00000000G  00  9E 00002    MOVAB   NML$AB_SNDBUFFER, R2                                    ;
                          62        04  AC  90 00009    MOVB    ERR, NML$AB_SNDBUFFER                                   ; 0603
                       01 A2        08  AC  B0 0000D    MOVW    DET, NML$AB_SNDBUFFER+1                                 ; 0604
                                        03  DD 00012    PUSHL   #3                                                      ; 0608
                                        52  DD 00014    PUSHL   R2                                                      ;
                             01F90000   8F  DD 00016    PUSHL   #33095680                                               ;
```

```
                    00000000G  00          03  FB 0001C          CALLS   #3, LIB$SIGNAL                    ;
                                               04 00023          RET                                      ; 0610

; Routine Size:  36 bytes,    Routine Base:  $CODE$ + 0209
```

```
;  619          0611  1 %SBTTL 'NML$DEBUG_TXT  Print text message'
;  620          0612  1 GLOBAL ROUTINE NML$DEBUG_TXT (BITNUM, TXTDSC) : NOVALUE =
;  621          0613  1
;  622          0614  1 !++
;  623          0615  1 ! FUNCTIONAL DESCRIPTION:
;  624          0616  1 !
;  625          0617  1 !       This routine prints the specified message text to SYS$OUTPUT if
;  626          0618  1 !       the appropriate logging flags are set.
;  627          0619  1 !
;  628          0620  1 ! FORMAL PARAMETERS:
;  629          0621  1 !
;  630          0622  1 !       BITNUM          Bit number of the logging flag.
;  631          0623  1 !       TXTDSC          Descriptor of ASCII text string.
;  632          0624  1 !
;  633          0625  1 ! IMPLICIT INPUTS:
;  634          0626  1 !
;  635          0627  1 !       NML$GL_LOGMASK  Values of current logging flags.
;  636          0628  1 !
;  637          0629  1 !--
;  638          0630  1
;  639          0631  2 BEGIN
;  640          0632  2
;  641          0633  2 MAP
;  642          0634  2     TXTDSC : REF DESCRIPTOR;
;  643          0635  2
;  644          0636  2 LITERAL
;  645          0637  2     FAOSIZE = 132;
;  646          0638  2
;  647          0639  2 LOCAL
;  648          0640  2     FAOPRM,
;  649          0641  2     OUTDSC : VECTOR [2],
;  650          0642  2     FAOBUF : BBLOCK [FAOSIZE];
;  651          0643  2
;  652          0644  2 !
;  653          0645  2 ! If the correct logging flag is set then output the text string.
;  654          0646  2 !
;  655          0647  2 IF .NML$GL_LOGMASK [.BITNUM]
;  656          0648  2 THEN
;  657          0649  3     BEGIN
;  658          0650  3     FAOPRM  = .TXTDSC;
;  659          0651  3     OUTDSC [0] = FAOSIZE;
;  660          0652  3     OUTDSC [1] = FAOBUF;
;  661        P 0653  3     $FAOL (CTRSTR = $ASCID ('!/*** !AS'),
;  662        P 0654  3            OUTLEN = OUTDSC [0],
;  663        P 0655  3            OUTBUF = OUTDSC,
;  664          0656  3            PRMLST = FAOPRM);
;  665          0657  3     LIB$PUT_OUTPUT (OUTDSC);
;  666          0658  2     END;
;  667          0659  2
;  668          0660  1 END;                                ! End of NML$DEBUG_TXT


                                                            .PSECT  $PLIT$,NOWRT,NOEXE,2

                    53  41  21  20  2A  2A  2A  2F  21   00002 P.AAC:  .ASCII  \!/*** !AS\                              ;
                                                         0000B        .BLKB   1
```

```
                                00000009  0000C P.AAB:  .LONG   9                                        ;
                                00000000' 00010         .ADDRESS P.AAC                                 ;

                                                        .EXTRN  SYS$FAOL

                                                        .PSECT  $CODE$,NOWRT,2

                                    0000 00000          .ENTRY  NML$DEBUG_TXT, Save nothing            ; 0612
                          5E     FF70   CE   9E 00002   MOVAB   -144(SP), SP                           ;
              2D 00000000G  00      04   AC   E1 00007   BBC     BITNUM, NML$GL_LOGMASK, 1$             ; 0647
                            6E      08   AC   D0 00010   MOVL    TXTDSC, FAOPRM                         ; 0650
                      F8    AD      84   8F   9A 00014   MOVZBL  #132, OUTDSC                           ; 0651
                      FC    AD      04   AE   9E 00019   MOVAB   FAOBUF, OUTDSC+4                       ; 0652
                                         5E   DD 0001E   PUSHL   SP                                     ; 0656
                                      F8 AD   9F 00020   PUSHAB  OUTDSC                                 ;
                                      F8 AD   9F 00023   PUSHAB  OUTDSC                                 ;
                                00000000' 00  9F 00026   PUSHAB  P.AAB                                  ;
                   00000000G  00            04   FB 0002C   CALLS   #4, SYS$FAOL                       ;
                                      F8 AD   9F 00033   PUSHAB  OUTDSC                                 ; 0657
                   00000000G  00            01   FB 00036   CALLS   #1, LIB$PUT_OUTPUT                 ;
                                               04 0003D 1$:  RET                                        ; 0660

; Routine Size:  62 bytes,    Routine Base:  $CODE$ + 022D
```

```
;  670     0661  1 %SBTTL 'NML$DEBUG_MSG  Print binary message'
;  671     0662  1 GLOBAL ROUTINE NML$DEBUG_MSG (BITNUM, BUFFER_ADR,
;  672     0663  1                                BUFFER_LEN, TXTDSC) : NOVALUE =
;  673     0664  1
;  674     0665  1 !++
;  675     0666  1 ! FUNCTIONAL DESCRIPTION:
;  676     0667  1 !
;  677     0668  1 !       This routine dumps binary messages to SYS$OUTPUT.
;  678     0669  1 !
;  679     0670  1 ! FORMAL PARAMETERS:
;  680     0671  1 !
;  681     0672  1 !       BITNUM          Number of the logging flag bit.
;  682     0673  1 !       BUFFER_ADR      Address of the message buffer.
;  683     0674  1 !       BUFFER_LEN      Length of the message in bytes.
;  684     0675  1 !       TXTDSC          Descriptor of text string.
;  685     0676  1 !
;  686     0677  1 ! IMPLICIT INPUTS:
;  687     0678  1 !
;  688     0679  1 !       NML$GL_LOGMASK  Values of current logging flags.
;  689     0680  1 !
;  690     0681  1 !--
;  691     0682  1
;  692     0683  2 BEGIN
;  693     0684  2
;  694     0685  2 MAP
;  695     0686  2     TXTDSC : REF DESCRIPTOR;
;  696     0687  2
;  697     0688  2 LITERAL
;  698     0689  2     FAOSIZ = 256,                       ! The print buffer.
;  699     0690  2     FAOLST_SIZE = 10,                   ! Size of FAO parameter vector
;  700     0691  2     DUMP_BUFFER_SIZE = 2000;
;  701     0692  2
;  702     0693  2 LOCAL
;  703     0694  2     FAOBUF   : VECTOR [FAOSIZ, BYTE],! Print buffer
;  704     0695  2     FAOLST   : VECTOR [FAOLST_SIZE],    ! List of args to $FAOL
;  705     0696  2     OUTDSC   : VECTOR [2],              ! Descriptor of the output line
;  706     0697  2     BYTES,                              ! Counter for bytes written
;  707     0698  2     PTR:        REF BBLOCK,
;  708     0699  2     I,                                  ! index
;  709     0700  2     BUFFER_END,                         ! Address of end of message buffer.
;  710     0701  2     DUMP_BUFFER :                       ! Buffer from which the data is dumped.
;  711     0702  2         BBLOCK [DUMP_BUFFER_SIZE];
;  712     0703  2
;  713     0704  2 !
;  714     0705  2 ! If the watcher logical name was set, and the information being logged is
;  715     0706  2 ! a NICE message, write the NICE message to the watcher's mailbox.
;  716     0707  2 !
;  717     0708  2 IF .BITNUM EQL DBG$C_NETIO AND
;  718     0709  2    .NML$GW_WATCHER_CHAN NEQ 0 THEN
;  719     0710  3     BEGIN
;  720     0711  3     $SETRWM (WATFLG = 1);       ! Don't wait if watcher's mailbox is full.
;  721   P 0712  3     $QIOW (FUNC = IO$_WRITEVBLK OR IO$M_NOW,
;  722   P 0713  3            CHAN = .NML$GW_WATCHER_CHAN,
;  723   P 0714  3            P1 = .BUFFER_ADR,
;  724     0715  3            P2 = .BUFFER_LEN);
;  725     0716  3     $SETRWM (WATFLG = 0);       ! Do wait for other resources.
;  726     0717  2     END;
```

```
;  727          0718  2 !
;  728          0719  2 ! If the correct logging flag is not set then just return.
;  729          0720  2 !
;  730          0721  2 IF NOT .NML$GL_LOGMASK [.BITNUM] THEN
;  731          0722  2     RETURN;
;  732          0723  2 !
;  733          0724  2 ! If the string length is nonzero then print it.
;  734          0725  2 !
;  735          0726  2 IF .TXTDSC NEQA 0 THEN
;  736          0727  3     BEGIN
;  737          0728  3
;  738          0729  3     OUTDSC [0] = FAOSIZ;
;  739          0730  3     OUTDSC [1] = FAOBUF;
;  740          0731  3
;  741          0732  3     FAOLST [0] = .TXTDSC [DSC$W_LENGTH];
;  742          0733  3     FAOLST [1] = .TXTDSC [DSC$A_POINTER];
;  743          0734  3     FAOLST [2] = .BUFFER_LEN;
;  744          0735  3
;  745        P 0736  3     $FAOL (CTRSTR = $ASCID ('!/ !AD  (length = !UL bytes)!/'),
;  746        P 0737  3                OUTLEN = OUTDSC [0],
;  747        P 0738  3                OUTBUF = OUTDSC,
;  748          0739  3                PRMLST = FAOLST);
;  749          0740  3
;  750          0741  3     LIB$PUT_OUTPUT (OUTDSC);
;  751          0742  3
;  752          0743  2     END;
;  753          0744  2 !
;  754          0745  2 ! Dumping permanent data base records requires BYPASS privilege because the
;  755          0746  2 ! passwords are displayed.
;  756          0747  2 !
;  757          0748  3 IF (.BITNUM EQL DBG$C_FILEIO)
;  758          0749  3     AND (NOT .NML$GQ_PROPRVMSK [PRV$V_BYPASS])
;  759          0750  2     AND (NOT .NML$GQ_PROPRVMSK [PRV$V_READALL]) THEN
;  760          0751  2     RETURN;
;  761          0752  2
;  762        . 0753  2 !
;  763          0754  2 ! Move the data to be dumped into the dump buffer, filling it with zeros.
;  764          0755  2 ! This ensures that any information past the end of the buffer is printed
;  765          0756  2 ! as zeros.
;  766          0757  2 !
;  767          0758  2 CH$COPY (.BUFFER_LEN, .BUFFER_ADR, 0, DUMP_BUFFER_SIZE, DUMP_BUFFER);
;  768          0759  2 !
;  769          0760  2 ! Dump the buffer contents in hex and ASCII.
;  770          0761  2 !
;  771          0762  2 OUTDSC [1] = FAOBUF;
;  772          0763  2 PTR = DUMP_BUFFER;
;  773          0764  2 BUFFER_END = DUMP_BUFFER + .BUFFER_LEN;
;  774          0765  2 WHILE .PTR LSS .BUFFER_END DO
;  775          0766  3     BEGIN
;  776          0767  3     OUTDSC [0] = FAOSIZ;
;  777          0768  3     FAOLST [0] = .PTR [12,0,32,0];
;  778          0769  3     FAOLST [1] = .PTR [8,0,32,0];
;  779          0770  3     FAOLST [2] = .PTR [4,0,32,0];
;  780          0771  3     FAOLST [3] = .PTR [0,0,32,0];
;  781          0772  3     FAOLST [4] = 16;
;  782          0773  3     FAOLST [5] = .PTR;
;  783        P 0774  3     $FAOL (CTRSTR = $ASCID ('!XL !XL !XL !XL !_!AF'),
```

```
;     P 0775   3               OUTLEN = OUTDSC [0],
;     P 0776   3               OUTBUF = OUTDSC,
;       0777   3               PRMLST = FAOLST);
;       0778   3       LIB$PUT_OUTPUT (OUTDSC);
;       0779   3       PTR = .PTR + 16;
;       0780   2       END;
;       0781   2   !
;       0782   2   ! Add a new line.
;       0783   2   !
;     P 0784   2   $FAOL    (CTRSTR = $ASCID ('!/'),
;     P 0785   2            OUTLEN = OUTDSC [0],
;     P 0786   2            OUTBUF = OUTDSC,
;       0787   2            PRMLST = 0);
;       0788   2   LIB$PUT_OUTPUT (OUTDSC);
;       0789   2
;       0790   1   END;                                      ! End of NML$DEBUG_MSG


                                                                    .PSECT  $PLIT$,NOWRT,NOEXE,2

68 74 67 6E 65 6C 28 20 20 44 41 21 20 2F 21  00014 P.AAE:  .ASCII  \!/ !AD  (length = !UL bytes)!/\        ;
2F 21 29 73 65 74 79 62 20 4C 55 21 20 3D 20  00023                                                     ;
                                              00032         .BLKB   2
                                    0000001E  00034 P.AAD:  .LONG   30                                  ;
                                    00000000' 00038         .ADDRESS P.AAE                              ;
4C 58 21 20 4C 58 21 20 4C 58 21 20 4C 58 21  0003C P.AAG:  .ASCII  \!XL !XL !XL !XL !_!AF\              ;
                        46 41 21 5F 21 20     0004B                                                     ;
                                              00051         .BLKB   3
                                    00000015  00054 P.AAF:  .LONG   21                                  ;
                                    00000000' 00058         .ADDRESS P.AAG                              ;
                                       2F 21  0005C P.AAI:  .ASCII  \!/\                                ;
                                              0005E         .BLKB   2
                                    00000002  00060 P.AAH:  .LONG   2                                   ;
                                    00000000' 00064         .ADDRESS P.AAI                              ;

                                                            .EXTRN  SYS$SETRWM, SYS$QIOW

                                                            .PSECT  $CODE$,NOWRT,2

                                        07FC  00000         .ENTRY  NML$DEBUG_MSG, Save R2,R3,R4,R5,R6,R7,R8,-  ; 0662
                                                                    R9,R10                              ;
                   5A 00000000G  00  9E  00002              MOVAB   NML$GW_WATCHER_CHAN, R10            ;
                   59 00000000G  00  9E  00009              MOVAB   SYS$SETRWM, R9                      ;
                   58 00000000G  00  9E  00010              MOVAB   LIB$PUT_OUTPUT, R8                  ;
                   57 00000000G  00  9E  00017              MOVAB   SYS$FAOL, R7                        ;
                   56 00000000'  00  9E  0001E              MOVAB   P.AAD, R6                           ;
                   5E      F700  CE  9E  00025              MOVAB   -2304(SP), SP                       ;
                   52        04  AC  D0  0002A              MOVL    BITNUM, R2                          ; 0708
                                 2A  12  0002E              BNEQ    1$                                  ;
                                 6A  B5  00030              TSTW    NML$GW_WATCHER_CHAN                 ; 0709
                                 26  13  00032              BEQL    1$                                  ;
                                 01  DD  00034              PUSHL   #1                                  ; 0711
                   69            01  FB  00036              CALLS   #1, SYS$SETRWM                      ;
                                 7E  7C  00039              CLRQ    -(SP)                               ; 0715
                                 7E  7C  0003B              CLRQ    -(SP)                               ;
                   7E        08  AC  7D  0003D              MOVQ    BUFFER_ADR, -(SP)                   ;
```

```
                                                 7E  7C 00041          CLRQ    -(SP)                                                  ;
                                                 7E  D4 00043          CLRL    -(SP)                                                  ;
                                  7E       70    8F  9A 00045          MOVZBL  #112, -(SP)                                            ;
                                  7E             6A  3C 00049          MOVZWL  NML$GW_WATCHER_CHAN, -(SP)                             ;
                                                 7E  D4 0004C          CLRL    -(SP)                                                  ;
                        00000000G 00             0C  FB 0004E          CALLS   #12, SYS$Q_IOW                                         ;
                                                 7E  D4 00055          CLRL    -(SP)                                                  ;  0716
                                  69             01  FB 00057          CALLS   #1, SYS$SETRWM                                         ;
                     01 00000000G 00             52  E0 0005A 1$:      BBS     R2, NML$GL_LOGMASK, 2$                                 ;  0721
                                                 04  00062              RET                                                            ;
                                  50       10    AC  D0 00063 2$:      MOVL    TXTDSC, R0                                             ;  0726
                                                 37  13 00067          BEQ     3$                                                     ;
                           FED0   CD       0100  8F  3C 00069          MOVZWL  #256, OUTDSC                                           ;  0729
                           FED4   CD       FF00  CD  9E 00070          MOVAB   FAOBUF, OUTDSC+4                                       ;  0730
                           FED8   CD             60  3C 00077          MOVZWL  (R0), FAOLST                                           ;  0732
                           FEDC   CD       04    A0  D0 0007C          MOVL    4(R0), FAOLST+4                                        ;  0733
                           FEE0   CD       0C    AC  D0 00082          MOVL    BUFFER_LEN, FAOLST+8                                   ;  0734
                                        FED8   CD  9F 00088          PUSHAB  FAOLST                                                 ;  0739
                                        FED0   CD  9F 0008C          PUSHAB  OUTDSC                                                 ;
                                        FED0   CD  9F 00090          PUSHAB  OUTDSC                                                 ;
                                               56  DD 00094          PUSHL   R6                                                     ;
                                  67           04  FB 00096          CALLS   #4, SYS$FAOL                                           ;
                                        FED0   CD  9F 00099          PUSHAB  OUTDSC                                                 ;  0741
                                  68           01  FB 0009D          CALLS   #1, LIB$PUT_OUTPUT                                     ;
                                  01           52  D1 000A0 3$:      CMPL    R2, #1                                                 ;  0748
                                               10  12 000A3          BNEQ    4$                                                     ;
                     08 00000000G 00             05  E0 000A5          BBS     #5, NML$GQ_PROPRVMSK+3, 4$                             ;  0749
                     7E 00000000G 00             03  E1 000AD          BBC     #3, NML$GQ_PROPRVMSK+4, 7$                             ;  0750
07D0    8F           00       08    BC    0C    AC  2C 000B5 4$:      MOVC5   BUFFER_LEN, @BUFFER_ADR, #0, #2000, -                  ;  0758
                                               6E  000BE              DUMP_BUFFER                                                    ;
                           FED4   CD       FF00  CD  9E 000BF          MOVAB   FAOBUF, OUTDSC+4                                       ;  0762
                                  52           6E  9E 000C6          MOVAB   DUMP_BUFFER, PTR                                       ;  0763
                                  50           6E  9E 000C9          MOVAB   DUMP_BUFFER, R0                                        ;  0764
53                                50       0C    AC  C1 000CC          ADDL3   BUFFER_LEN, R0, BUFFER_END                             ;
                                  53           52  D1 000D1 5$:      CMPL    PTR, BUFFER_END                                        ;  0765
                                               46  18 000D4          BGEQ    6$                                                     ;
                           FED0   CD       0100  8F  3C 000D6          MOVZWL  #256, OUTDSC                                           ;  0767
                           FED8   CD       0C    A2  D0 000DD          MOVL    12(PTR), FAOLST                                        ;  0768
                           FEDC   CD       08    A2  D0 000E3          MOVL    8(PTR), FAOLST+4                                       ;  0769
                           FEE0   CD       04    A2  D0 000E9          MOVL    4(PTR), FAOLST+8                                       ;  0770
                           FEE4   CD             62  D0 000EF          MOVL    (PTR), FAOLST+12                                       ;  0771
                           FEE8   CD             10  D0 000F4          MOVL    #16, FAOLST+16                                         ;  0772
                           FEEC   CD             52  D0 000F9          MOVL    PTR, FAOLST+20                                         ;  0773
                                        FED8   CD  9F 000FE          PUSHAB  FAOLST                                                 ;  0777
                                        FED0   CD  9F 00102          PUSHAB  OUTDSC                                                 ;
                                        FED0   CD  9F 00106          PUSHAB  OUTDSC                                                 ;
                                          20   A6  9F 0010A          PUSHAB  P.AAF                                                  ;
                                  67           04  FB 0010D          CALLS   #4, SYS$FAOL                                           ;
                                        FED0   CD  9F 00110          PUSHAB  OUTDSC                                                 ;  0778
                                  68           01  FB 00114          CALLS   #1, LIB$PUT_OUTPUT                                     ;
                                  52           10  C0 00117          ADDL2   #16, PTR                                               ;  0779
                                               B5  11 0011A          BRB     5$                                                     ;  0765
                                               7E  D4 0011C 6$:      CLRL    -(SP)                                                  ;  0787
                                        FED0   CD  9F 0011E          PUSHAB  OUTDSC                                                 ;
                                        FED0   CD  9F 00122          PUSHAB  OUTDSC                                                 ;
                                          2C   A6  9F 00126          PUSHAB  P.AAH                                                  ;
                                  67           04  FB 00129          CALLS   #4, SYS$FAOL                                           ;
```

```
                                FEDO    CD  9F 0012C           PUSHAB  OUTDSC                          ; 0788
                                    68  01  FB 00130           CALLS   #1, LIB$PUT_OUTPUT             ;
                                        04 00133 7$:           RET                                    ; 0790

; Routine Size:  308 bytes,    Routine Base:  $CODE$ + 026B
```

```
;  801          0791  1 %SBTTL 'NML$DEBUG_QIO  Print NETACP QIO information'
;  802          0792  1 GLOBAL ROUTINE NML$DEBUG_QIO (BITNUM, QIOS, IOSB, P1DSC,
;  803          0793  1                               P2DSC, P3ADR, P4DSC, TXTDSC) : NOVALUE =
;  804          0794  1
;  805          0795  1 !++
;  806          0796  1 ! FUNCTIONAL DESCRIPTION:
;  807          0797  1 !
;  808          0798  1 !       This routine dumps NETACP QIO information to SYS$OUTPUT.
;  809          0799  1 !
;  810          0800  1 ! FORMAL PARAMETERS:
;  811          0801  1 !
;  812          0802  1 !       BITNUM          Contains the number of the logging flag bit.
;  813          0803  1 !       QIOS            Status of QIO (R0).
;  814          0804  1 !       IOSB            Address of I/O status block.
;  815          0805  1 !       P1DSC           Address of P1 descriptor.
;  816          0806  1 !       P2DSC           Address of P2 descriptor.
;  817          0807  1 !       P3ADR           Address of P3 word.
;  818          0808  1 !       P4DSC           Address of P4 descriptor.
;  819          0809  1 !       TXTDSC          Descriptor of text string.
;  820          0810  1 !
;  821          0811  1 ! IMPLICIT INPUTS:
;  822          0812  1 !
;  823          0813  1 !       NML$GL_LOGMASK  Values of current logging flags.
;  824          0814  1 !
;  825          0815  1 !--
;  826          0816  1
;  827          0817  2 BEGIN
;  828          0818  2
;  829          0819  2 MAP
;  830          0820  2         IOSB    : REF $IOSB,
;  831          0821  2         P1DSC : REF DESCRIPTOR,
;  832          0822  2         P2DSC   : REF DESCRIPTOR,
;  833          0823  2         P4DSC   : REF DESCRIPTOR;
;  834          0824  2
;  835          0825  2 BIND
;  836        P 0826  2         FAOSTR = $ASCID ('!/R0=!XL IOSB=!XL/!XL NFB=!XW/!XL',
;  837          0827  2                          '!/P2=!XW/!XL P3=!XL (!XW) P4=!XW/!XL');
;  838          0828  2
;  839          0829  2 LITERAL
;  840          0830  2         FAOSIZ = 256;                   ! The print buffer
;  841          0831  2
;  842          0832  2 LOCAL
;  843          0833  2         FAOBUF : VECTOR [FAOSIZ, BYTE], ! Print buffer
;  844          0834  2         FAOLST : VECTOR [20],           ! List of args to $FAOL
;  845          0835  2         OUTDSC : VECTOR [2],            ! Descriptor of the output line
;  846          0836  2         BUFSIZ;
;  847          0837  2 !
;  848          0838  2 ! If the correct logging flag is not enabled then just return.
;  849          0839  2 !
;  850          0840  2 IF NOT .NML$GL_LOGMASK [.BITNUM]
;  851          0841  2 THEN
;  852          0842  2         RETURN;
;  853          0843  2
;  854          0844  2 !
;  855          0845  2 ! Print header message at beginning of QIO information.
;  856          0846  2 !
;  857          0847  2 IF .TXTDSC NEQ 0 THEN
```

```
;  858     0848  2      NML$DEBUG_TXT (.BITNUM, .TXTDSC);
;  859     0849  2
;  860     0850  2  OUTDSC [0] = FAOSIZ;
;  861     0851  2  OUTDSC [1] = FAOBUF;
;  862     0852  2  !
;  863     0853  2  ! Log the QIO completion status, IOSB, and the values of the QIO
;  864     0854  2  ! parameters.
;  865     0855  2  !
;  866     0856  2  FAOLST [0] = .QIOS;
;  867     0857  2  FAOLST [1] = .IOSB [0,0,32,0];
;  868     0858  2  FAOLST [2] = .IOSB [4,0,32,0];
;  869     0859  2
;  870     0860  2  IF .P1DSC NEQA 0 THEN
;  871     0861  3      BEGIN
;  872     0862  3      FAOLST [3] = .P1DSC [DSC$W_LENGTH];
;  873     0863  3      FAOLST [4] = .P1DSC [DSC$A_POINTER];
;  874     0864  3      END
;  875     0865  2  ELSE
;  876     0866  3      BEGIN
;  877     0867  3      FAOLST [3] = 0;
;  878     0868  3      FAOLST [4] = 0;
;  879     0869  2      END;
;  880     0870  2
;  881     0871  2  IF .P2DSC NEQA 0
;  882     0872  2  THEN
;  883     0873  3          BEGIN
;  884     0874  3
;  885     0875  3          FAOLST [5] = .P2DSC [DSC$W_LENGTH];
;  886     0876  3          FAOLST [6] = .P2DSC [DSC$A_POINTER];
;  887     0877  3
;  888     0878  3          END
;  889     0879  2  ELSE
;  890     0880  3          BEGIN
;  891     0881  3
;  892     0882  3          FAOLST [5] = 0;
;  893     0883  3          FAOLST [6] = 0;
;  894     0884  3
;  895     0885  2          END;
;  896     0886  2
;  897     0887  2  FAOLST [7] = .P3ADR;
;  898     0888  2  IF .P3ADR NEQA 0
;  899     0889  2  THEN
;  900     0890  2          FAOLST [8] = .(.P3ADR)<0,16>
;  901     0891  2  ELSE
;  902     0892  2          FAOLST [8] = 0;
;  903     0893  2
;  904     0894  2  IF .P4DSC NEQA 0
;  905     0895  2  THEN
;  906     0896  3          BEGIN
;  907     0897  3
;  908     0898  3          FAOLST [9] = .P4DSC [DSC$W_LENGTH];
;  909     0899  3          FAOLST [10] = .P4DSC [DSC$A_POINTER];
;  910     0900  3
;  911     0901  3          END
;  912     0902  2  ELSE
;  913     0903  3          BEGIN
;  914     0904  3
```

```
;  915          0905  3          FAOLST [9] = 0;
;  916          0906  3          FAOLST [10] = 0;
;  917          0907  3
;  918          0908  2          END;
;  919          0909  2
;  920        P 0910  2 $FAOL (CTRSTR = FAOSTR,
;  921        P 0911  2             OUTLEN = OUTDSC [0],
;  922        P 0912  2             OUTBUF = OUTDSC,
;  923          0913  2             PRMLST = FAOLST);
;  924          0914  2
;  925          0915  2 LIB$PUT_OUTPUT (OUTDSC);                ! Write to SYS$OUTPUT
;  926          0916  2
;  927          0917  2 OUTDSC [0] = 0;
;  928          0918  2 OUTDSC [1] = FAOBUF;
;  929          0919  2 BUFSIZ = FAOSIZ;
;  930          0920  2 IF NOT .QIOS
;  931          0921  2 THEN
;  932          0922  2          NML$ADD_MSG_TXT (.QIOS, BUFSIZ, OUTDSC [0], .OUTDSC [1])
;  933          0923  2 ELSE
;  934          0924  2          NML$ADD_MSG_TXT (.IOSB [IOS$W_STATUS],
;  935          0925  2                           BUFSIZ,
;  936          0926  2                           OUTDSC [0],
;  937          0927  2                           .OUTDSC [1]);
;  938          0928  2
;  939          0929  2 LIB$PUT_OUTPUT (OUTDSC);                ! Write to SYS$OUTPUT
;  940          0930  2
;  941          0931  2 !
;  942          0932  2 ! Dump the contents of the NFB, the P2 (Key) buffer, and the P4 (Value) buffer.
;  943          0933  2 !
;  944          0934  2 NML$DUMP_QIO_BUFS (.BITNUM, .P1DSC, .P2DSC, .P4DSC, .P3ADR);
;  945          0935  2
;  946          0936  1 END;                                    ! End of NML$DEBUG_QIO


                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

21  3D  42  53  4F  49  20  4C  58  21  3D  30  52  2F  21  00068 P.AAK:  .ASCII  \!/R0=!XL IOSB=!XL/!XL NFB=!XW/!XL!/P2=!X\  ;
2F  57  58  21  3D  42  46  4E  20  4C  58  21  2F  4C  58  00077                                                            ;
                58  21  3D  32  50  2F  21  4C  58  21      00086                                                            ;
21  28  20  4C  58  21  3D  33  50  20  4C  58  21  2F  57  00090          .ASCII  \W/!XL P3=!XL (!XW) P4=!XW/!XL\           ;
    4C  58  21  2F  57  58  21  3D  34  50  20  29  57  58  0009F                                                            ;
                                                    000AD   .BLKB   3
                                        00000045    000B0 P.AAJ:  .LONG   69                                                 ;
                                        00000000'   000B4         .ADDRESS P.AAK                                         ;

                                                        FAOSTR=             P.AAJ


                                                                .PSECT  $CODE$,NOWRT,2

                                                  007C 00000            .ENTRY  MML$DEBUG_QIO, Save R2,R3,R4,R5,R6          ; 0792
                                  56 00000000G    00   9E 00002         MOVAB   LIB$PUT_OUTPUT, R6                          ;
                                  5E      FEA4    CE   9E 00009         MOVAB   -348(SP), SP                                ;
                      01 00000000G  00      04    AC   E0 0000E         BBS     BITNUM, NML$GL_LOGMASK, 1$                  ; 0840
                                                    04 00017            RET                                                 ;
                                            20    AC   D5 00018 1$:     TSTL    TXTDSC                                      ; 0847
```

```
                                 0B  13 0001B          BEQL    2$                                  ;
                         20      AC  DD 0001D          PUSHL   TXTDSC                              ; 0848
                         04      AC  DD 00020          PUSHL   BITNUM                              ;
        FE66    CF               02  FB 00023          CALLS   #2, NML$DEBUG_TXT                   ;
          04    AE     0100      8F  3C 00028 2$:      MOVZWL  #256, OUTDSC                        ; 0850
          08    AE       5C      AE  9E 0002E          MOVAB   FAOBUF, OUTDSC+4                    ; 0851
          0C    AE       08      AC  D0 00033          MOVL    QIOS, FAOLST                        ; 0856
                55       0C      AC  D0 00038          MOVL    IOSB, R5                            ; 0857
          10    AE               65  7D 0003C          MOVQ    (R5), FAOLST+4                      ;
                54       10      AC  D0 00040          MOVL    P1DSC, R4                           ; 0860
                                 0B  13 00044          BEQL    3$                                  ;
          18    AE               64  3C 00046          MOVZWL  (R4), FAOLST+12                     ; 0862
          1C    AE       04      A4  D0 0004A          MOVL    4(R4), FAOLST+16                    ; 0863
                                 03  11 0004F          BRB     4$                                  ; 0860
                         18      AE  7C 00051 3$:      CLRQ    FAOLST+12                           ; 0867
                53       14      AC  D0 00054 4$:      MOVL    P2DSC, R3                           ; 0871
                                 0B  13 00058          BEQL    5$                                  ;
          20    AE               63  3C 0005A          MOVZWL  (R3), FAOLST+20                     ; 0875
          24    AE       04      A3  D0 0005E          MOVL    4(R3), FAOLST+24                    ; 0876
                                 03  11 00063          BRB     6$                                  ; 0871
                         20      AE  7C 00065 5$:      CLRQ    FAOLST+20                           ; 0882
          28    AE       18      AC  D0 00068 6$:      MOVL    P3ADR, FAOLST+28                    ; 0887
                                 07  13 0006D          BEQL    7$                                  ; 0888
          2C    AE       18      BC  3C 0006F          MOVZWL  @P3ADR, FAOLST+32                   ; 0890
                                 03  11 00074          BRB     8$                                  ;
                         2C      AE  D4 00076 7$:      CLRL    FAOLST+32                           ; 0892
                52       1C      AC  D0 00079 8$:      MOVL    P4DSC, R2                           ; 0894
                                 0B  13 0007D          BEQL    9$                                  ;
          30    AE               62  3C 0007F          MOVZWL  (R2), FAOLST+36                     ; 0898
          34    AE       04      A2  D0 00083          MOVL    4(R2), FAOLST+40                    ; 0899
                                 03  11 00088          BRB     10$                                 ; 0894
                         30      AE  7C 0008A 9$:      CLRQ    FAOLST+36                           ; 0905
                         0C      AE  9F 0008D 10$:     PUSHAB  FAOLST                              ; 0913
                         08      AE  9F 00090          PUSHAB  OUTDSC                              ;
                         0C      AE  9F 00093          PUSHAB  OUTDSC                              ;
                  00000000'      00  9F 00096          PUSHAB  FAOSTR                              ;
   00000000G    00               04  FB 0009C          CALLS   #4, SYS$FAOL                        ;
                         04      AE  9F 000A3          PUSHAB  OUTDSC                              ; 0915
                66               01  FB 000A6          CALLS   #1, LIB$PUT_OUTPUT                  ;
                         04      AE  D4 000A9          CLRL    OUTDSC                              ; 0917
          08    AE       5C      AE  9E 000AC          MOVAB   FAOBUF, OUTDSC+4                    ; 0918
                6E     0100      8F  3C 000B1          MOVZWL  #256, BUFSIZ                        ; 0919
                0E       08      AC  E8 000B6          BLBS    QIOS, 11$                           ; 0922
                         08      AE  DD 000BA          PUSHL   OUTDSC+4                            ;
                         08      AE  9F 000BD          PUSHAB  OUTDSC                              ;
                         08      AE  9F 000C0          PUSHAB  BUFSIZ                              ;
                         08      AC  DD 000C3          PUSHL   QIOS                                ;
                                 0C  11 000C6          BRB     12$                                 ;
                         08      AE  DD 000C8 11$:     PUSHL   OUTDSC+4                            ; 0927
                         08      AE  9F 000CB          PUSHAB  OUTDSC                              ; 0926
                         08      AE  9F 000CE          PUSHAB  BUFSIZ                              ; 0924
                7E               65  3C 000D1          MOVZWL  (R5), -(SP)                         ;
        FC6D    CF               04  FB 000D4 12$:     CALLS   #4, NML$ADD_MSG_TXT                 ;
                         04      AE  9F 000D9          PUSHAB  OUTDSC                              ; 0929
                66               01  FB 000DC          CALLS   #1, LIB$PUT_OUTPUT                  ;
                         18      AC  DD 000DF          PUSHL   P3ADR                               ; 0934
                                 52  DD 000E2          PUSHL   R2                                  ;
```

```
                                    53  DD 000E4         PUSHL   R3                                      ;
                                    54  DD 000E6         PUSHL   R4                                      ;
                                04  AC  DD 000E8         PUSHL   BITNUM                                  ;
                  00000000V  00         05  FB 000EB     CALLS   #5, NML$DUMP_QIO_BUFS                   ;
                                        04 000F2         RET                                             ; 0936

; Routine Size:  243 bytes,    Routine Base:  $CODE$ + 039F
```

```
;  948          0937  1
;  949          0938  1 %SBTTL 'NML$DUMP_QIO_BUFS  Dump QIO buffers'
;  950          0939  1 GLOBAL ROUTINE NML$DUMP_QIO_BUFS (BITNUM, P1DSC, P2DSC, P4DSC, P3ADR) :
;  951          0940  1                                                 NOVALUE =
;  952          0941  1
;  953          0942  2 BEGIN
;  954          0943  2
;  955          0944  2 LOCAL
;  956          0945  2         P4LEN;          ! Length of P4 buffer
;  957          0946  2
;  958          0947  2 MAP
;  959          0948  2         P1DSC   : REF DESCRIPTOR,
;  960          0949  2         P2DSC   : REF DESCRIPTOR,
;  961          0950  2         P4DSC   : REF DESCRIPTOR;
;  962          0951  2
;  963          0952  2 IF .P1DSC NEQ 0 THEN
;  964          0953  2     NML$DEBUG_MSG ( .BITNUM,
;  965          0954  2                     .P1DSC [DSC$A_POINTER],
;  966          0955  2                     .P1DSC [DSC$W_LENGTH],
;  967          0956  2                     $ASCID('P1 buffer contents'));
;  968          0957  2
;  969          0958  2 IF .P2DSC NEQ 0
;  970          0959  2 THEN
;  971          0960  2     NML$DEBUG_MSG ( .BITNUM,
;  972          0961  2                     .P2DSC [DSC$A_POINTER],
;  973          0962  2                     .P2DSC [DSC$W_LENGTH],
;  974          0963  2                     $ASCID ('P2 buffer contents'));
;  975          0964  2
;  976          0965  2 IF .P4DSC NEQ 0
;  977          0966  2 THEN
;  978          0967  3     BEGIN
;  979          0968  3     !
;  980          0969  3     ! Figure out how much of the P4 buffer to dump.  If it's a
;  981          0970  3     ! show, the byte count was returned in P3.  If it's a set,
;  982          0971  3     ! the byte count is in the P4 buffer descriptor.
;  983          0972  3     !
;  984          0973  3     IF .P3ADR NEQ 0 THEN
;  985          0974  3         IF .(.P3ADR)<0,16> GTR NML$K_QIOBFLEN THEN
;  986          0975  3             P4LEN = 64
;  987          0976  3         ELSE
;  988          0977  3             P4LEN = .(.P3ADR)<0,16>
;  989          0978  3     ELSE
;  990          0979  3         P4LEN = .P4DSC [DSC$W_LENGTH];
;  991          0980  3     NML$DEBUG_MSG ( .BITNUM,
;  992          0981  3                     .P4DSC [DSC$A_POINTER],
;  993          0982  3                     .P4LEN,
;  994          0983  3                     $ASCID ('P4 buffer contents'));
;  995          0984  2     END;
;  996          0985  1 END;    ! of  NML$DUMP_QIO_BUFS


                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

65  74  6E  6F  63  20  72  65  66  66  75  62  20  31  50  000B8 P.AAM:  .ASCII  \P1 buffer contents\                    ;
                                                    73  74  6E  000C7                                                  ;
                                                                000CA          .BLKB   2
```

```
                                                00000012  000CC P.AAL:  .LONG   18                                        ;
                                                00000000'  000D0        .ADDRESS P.AAM                                   ;
65  74  6E  6F  63  20  72  65  66  66  75  62  20  32  50  000D4 P.AAO:  .ASCII  \P2 buffer contents\                      ;
                                                    73  74  6E  000E3                                                    ;
                                                            000E6        BLKB   2                                        ;
                                                00000012  000E8 P.AAN:  .LONG   18                                        ;
                                                00000000'  000EC        .ADDRESS P.AAO                                   ;
65  74  6E  6F  63  20  72  65  66  66  75  62  20  34  50  000F0 P.AAQ:  .ASCII  \P4 buffer contents\                      ;
                                                    73  74  6E  000FF                                                    ;
                                                            00102        .BLKB   2                                       ;
                                                00000012  00104 P.AAP:  .LONG   18                                        ;
                                                00000000'  00108        .ADDRESS P.AAQ                                   ;


                                                                        .PSECT  $CODE$,NOWRT,2

                                                    000C 00000          .ENTRY  NML$DUMP_QIO_BUFS, Save R2,R3            ; 0939
                                    53 00000000'  00  9E 00002          MOVAB   P.AAL, R3                                ;
                                    52      FDCC  CF  9E 00009          MOVAB   NML$DEBUG_MSG, R2                        ;
                                    50        08  AC  D0 0000E          MOVL    P1DSC, R0                                ; 0952
                                                  0E  13 00012          BEQL    1$                                       ;
                                                  53  DD 00014          PUSHL   R3                                       ; 0956
                                    7E            60  3C 00016          MOVZWL  (R0), -(SP)                              ; 0955
                                              04  A0  DD 00019          PUSHL   4(R0)                                    ; 0954
                                              04  AC  DD 0001C          PUSHL   BITNUM                                   ; 0953
                                    62            04  FB 0001F          CALLS   #4, NML$DEBUG_MSG                        ;
                                    50        0C  AC  D0 00022 1$:      MOVL    P2DSC, R0                                ; 0958
                                                  0F  13 00026          BEQL    2$                                       ;
                                              1C  A3  9F 00028          PUSHAB  P.AAN                                    ; 0963
                                    7E            60  3C 0002B          MOVZWL  (R0), -(SP)                              ; 0962
                                              04  A0  DD 0002E          PUSHL   4(R0)                                    ; 0961
                                              04  AC  DD 00031          PUSHL   BITNUM                                   ; 0960
                                    62            04  FB 00034          CALLS   #4, NML$DEBUG_MSG                        ;
                                    51        10  AC  D0 00037 2$:      MOVL    P4DSC, R1                                ; 0965
                                                  2A  13 0003B          BEQL    6$                                       ;
                                              14  AC  D5 0003D          TSTL    P3ADR                                    ; 0973
                                                  14  13 00040          BEQL    4$                                       ;
                              04B0  8F            14  BC  B1 00042      CMPW    @P3ADR, #1200                            ; 0974
                                                  06  1B 00048          BLEQU   3$                                       ;
                                    50        40  8F  9A 0004A          MOVZBL  #64, P4LEN                               ; 0975
                                                  09  11 0004E          BRB     5$                                       ;
                                    50        14  BC  3C 00050 3$:      MOVZWL  @P3ADR, P4LEN                            ; 0977
                                                  03  11 00054          BRB     5$                                       ; 0974
                                    50            61  3C 00056 4$:      MOVZWL  (R1), P4LEN                              ; 0979
                                              38  A3  9F 00059 5$:      PUSHAB  P.AAP                                    ; 0983
                                                  50  DD 0005C          PUSHL   P4LEN                                    ; 0982
                                              04  A1  DD 0005E          PUSHL   4(R1)                                    ; 0981
                                              04  AC  DD 00061          PUSHL   BITNUM                                   ; 0980
                                    62            04  FB 00064          CALLS   #4, NML$DEBUG_MSG                        ;
                                                  04  00067 6$:        RET                                              ; 0985

; Routine Size:  104 bytes,    Routine Base:  $CODE$ + 0492
```

```
;  998          0986  1 %SBTTL 'NML$LOGALLPDB  Log contents of permanent data base files'
;  999          0987  1 GLOBAL ROUTINE NML$LOGALLPDB : NOVALUE =
; 1000          0988  1
; 1001          0989  1 !++
; 1002          0990  1 ! FUNCTIONAL DESCRIPTION:
; 1003          0991  1 !
; 1004          0992  1 !       This routine logs the contents of all permanent data base files
; 1005          0993  1 !       according to the setting of the logging bits.
; 1006          0994  1 !
; 1007          0995  1 !--
; 1008          0996  1
; 1009          0997  2     BEGIN
; 1010          0998  2
; 1011          0999  2     OWN
; 1012          1000  2         TAB : VECTOR [NMA$C_OPN_MAX - NMA$C_OPN_MIN + 1]
; 1013          1001  2               INITIAL (DBG$C_DMPNOD,
; 1014          1002  2                        DBG$C_DMPLIN,
; 1015          1003  2                        DBG$C_DMPLOG,
; 1016          1004  2                        DBG$C_DMPOBJ,
; 1017          1005  2                        DBG$C_DMPCIR,
; 1018          1006  2                        DBG$C_DMPX25,
; 1019          1007  2                        DBG$C_DMPX29,
; 1020          1008  2                        DBG$C_DMPCNF);
; 1021          1009  2
; 1022          1010  2     INCR IDX FROM NMA$C_OPN_MIN TO NMA$C_OPN_MAX DO
; 1023          1011  3         BEGIN
; 1024          1012  3
; 1025          1013  3         IF .NML$GL_LOGMASK [.TAB[.IDX]] THEN
; 1026          1014  4                 BEGIN
; 1027          1015  4                 IF NMA$OPENFILE (.IDX, NMA$C_OPN_AC_RO) THEN
; 1028          1016  5                         BEGIN
; 1029          1017  5                         NML$FILEDMP (.TAB [.IDX], .IDX);
; 1030          1018  5                         NML$CLOSEFILE (.IDX);
; 1031          1019  4                         END;
; 1032          1020  3                 END;
; 1033          1021  2         END;
; 1034          1022  2
; 1035          1023  1     END;                                    ! End of NML$LOGALLPDB


                                                        .PSECT  $OWN$,NOEXE,2

                                                000FF   .BLKB   1
00000015  00000014  00000013  00000012  00000011  00000010  00100 TAB:    .LONG   16, 17, 18, 19, 20, 21, 22, 23          ;
                                        00000017  00000016  00118                                                       ;


                                                        .PSECT  $CODE$,NOWRT,2

                                              000C 00000        .ENTRY  NML$LOGALLPDB, Save R2,R3                       ; 0987
                                           52   D4 00002        CLRL    IDX                                             ; 1010
                                53 00000000'0042   DE 00004 1$: MOVAL   TAB[IDX], R3                                   ; 1013
                     22 00000000G   00             63   E1 0000C        BBC     (R3), NML$GL_LOGMASK, 2$                ;
                                           7E   D4 00014        CLRL    -(SP)                                           ; 1015
                                           52   DD 00016        PUSHL   IDX                                             ;
```

```
                    00000000G  00          02  FB 00018          CALLS   #2, NMA$OPENFILE                         ;
                               14          50  E9 0001F          BLBC    R0, 2$                                  ;
                                           52  DD 00022          PUSHL   IDX                                     ; 1017
                                           63  DD 00024          PUSHL   (R3)                                    ;
                    00000000V  00          02  FB 00026          CALLS   #2, NML$FILEDMP                         ;
                                           52  DD 0002D          PUSHL   IDX                                     ; 1018
                    00000000G  00          01  FB 0002F          CALLS   #1, NML$CLOSEFILE                       ;
                          CA               52  07  F3 00036 2$:   AOBLEQ  #7, IDX, 1$                           ; 1010
                                               04  0003A          RET                                             ; 1023

; Routine Size:  59 bytes,    Routine Base:  $CODE$ + 04FA
```

```
;  1037     1024  1 %SBTTL 'NML$FILEDMP  Dump all the records in the file'
;  1038     1025  1 GLOBAL ROUTINE NML$FILEDMP (BITNUM, FID) : NOVALUE =
;  1039     1026  1
;  1040     1027  1 !++
;  1041     1028  1 ! FUNCTIONAL DESCRIPTION:
;  1042     1029  1 !
;  1043     1030  1 !
;  1044     1031  1 ! FORMAL PARAMETERS:
;  1045     1032  1 !
;  1046     1033  1 !       BITNUM          Logging flag bit number.
;  1047     1034  1 !       FID             Permanent data base file identification code.
;  1048     1035  1 !
;  1049     1036  1 ! IMPLICIT INPUTS:
;  1050     1037  1 !
;  1051     1038  1 !       NML$GL_LOGMASK  Values of current logging flags.
;  1052     1039  1 !
;  1053     1040  1 ! IMPLICIT OUTPUTS:
;  1054     1041  1 !
;  1055     1042  1 !
;  1056     1043  1 ! ROUTINE VALUE:
;  1057     1044  1 ! COMPLETION CODES:
;  1058     1045  1 !
;  1059     1046  1 !       NONE
;  1060     1047  1 !
;  1061     1048  1 ! SIDE EFFECTS:
;  1062     1049  1 !
;  1063     1050  1 !       NONE
;  1064     1051  1 !
;  1065     1052  1 !--
;  1066     1053  1
;  1067     1054  2     BEGIN
;  1068     1055  2
;  1069     1056  2     LITERAL
;  1070     1057  2         FAOSIZ = 256;                   ! The print buffer
;  1071     1058  2
;  1072     1059  2     LOCAL
;  1073     1060  2         FAOBUF : VECTOR [FAOSIZ, BYTE], ! Print buffer
;  1074     1061  2         FAOLST : VECTOR [100],          ! List of args to $FAOL
;  1075     1062  2         OUTDSC : DESCRIPTOR,            ! Descriptor of the output line
;  1076     1063  2         SAVEFLAG,                       ! Temporary file I/O logging flag
;  1077     1064  2         KEY    : WORD,                  ! Temporary record key buffer
;  1078     1065  2         RECDSC : DESCRIPTOR;            ! Record descriptor
;  1079     1066  2 !
;  1080     1067  2 ! If logging for this file is not enabled then just return.
;  1081     1068  2 !
;  1082     1069  2     IF NOT .NML$GL_LOGMASK [.BITNUM]
;  1083     1070  2     THEN
;  1084     1071  2         RETURN;                                                          ; Rc
;  1085     1072  2 !
;  1086     1073  2 ! Output the header.
;  1087     1074  2 !                                                                       ; 14
;  1088     1075  2     OUTDSC [DSC$W_LENGTH] = FAOSIZ;
;  1089     1076  2     OUTDSC [DSC$A_POINTER] = FAOBUF;
;  1090     1077  2
;  1091     1078  2     NML$APPENDTXT (.FID,
;  1092     1079  2                    $ASCID ('******************************'),
;  1093     1080  2                    $ASCID ('file dump ******************************'),
```

```
;  1094          1081  2                    OUTDSC,
;  1095          1082  2                    OUTDSC [DSC$W_LENGTH]);
;  1096          1083  2
;  1097          1084  2    LIB$PUT_OUTPUT (OUTDSC);
;  1098          1085  2
;  1099          1086  2  ! Save the value of the file I/O logging flag and set it to enable records
;  1100          1087  2  ! to be logged.
;  1101          1088  2  !
;  1102          1089  2    SAVEFLAG = .NML$GL_LOGMASK [DBG$C_FILEIO];
;  1103          1090  2    NML$GL_LOGMASK [DBG$C_FILEIO] = 1;
;  1104          1091  2
;  1105          1092  2  ! Read all records.
;  1106          1093  2  !
;  1107          1094  2    KEY = 0;                                ! Initialize record key
;  1108          1095  2    WHILE NMA$READREC (.FID, KEY, NML$GQ_RECBFDSC, RECDSC) DO
;  1109          1096  3        BEGIN
;  1110          1097  3
;  1111          1098  3        KEY = .KEY + 1;
;  1112          1099  3
;  1113          1100  2        END;
;  1114          1101  2
;  1115          1102  2  ! Output the trailer.
;  1116          1103  2  !
;  1117          1104  2    OUTDSC [DSC$W_LENGTH] = FAOSIZ;
;  1118          1105  2    OUTDSC [DSC$A_POINTER] = FAOBUF;
;  1119          1106  2
;  1120          1107  2    NML$APPENDTXT (.FID,
;  1121          1108  2                   $ASCID ('************************** End of'),
;  1122          1109  2                   $ASCID ('file dump ***************************'),
;  1123          1110  2                   OUTDSC,
;  1124          1111  2                   OUTDSC [DSC$W_LENGTH]);
;  1125          1112  2
;  1126          1113  2    LIB$PUT_OUTPUT (OUTDSC);
;  1127          1114  2
;  1128          1115  2  ! Restore the setting of the file I/O logging flag.
;  1129          1116  2  !
;  1130          1117  2    NML$GL_LOGMASK [DBG$C_FILEIO] = .SAVEFLAG;
;  1131          1118  2
;  1132          1119  1    END;                                    ! End of NML$FILEDMP


                                                                    .PSECT  $PLIT$,NOWRT,NOEXE,2

2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  0010C P.AAS:   .ASCII  \*****************************\          ;
    2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A      0011B                                                    ;
                                                            00129          .BLKB   3                                    ;
                                                  0000001D  0012C P.AAR:   .LONG   29                                   ;
                                                  00000000' 00130          .ADDRESS P.AAS                               ;
2A  2A  2A  2A  2A  20  70  6D  75  64  20  65  6C  69  66  00134 P.AAU:   .ASCII  \file dump *****************************\ ;
2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  00143                                                        ;
                    2A  2A  2A  2A  2A  2A  2A  2A  2A  00152                                                        ;
                                                            0015B          .BLKB   1                                    ;
                                                  00000027  0015C P.AAT:   .LONG   39                                   ;
                                                  00000000' 00160          .ADDRESS P.AAU                               ;
2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  00164 P.AAW:   .ASCII  \************************** End of\  ;
64  6E  45  20  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  00173                                                        ;
```

```
                                    66  6F  20  00182                                                              ;
                                                00185          .BLKB    3
                                      00000021  00188 P.AAV:   .LONG    33                                         ;
                                     00000000'  0018C          .ADDRESS P.AAW                                      ;
2A  2A  2A  2A  2A  20  70  6D  75  64  20  65  6C  69  66  00190 P.AAY:   .ASCII   \file dump **************************\  ;
2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  0019F                                                   ;
                                2A  2A  2A  2A  2A  2A  001AE                                                   ;
                                      00000024  001B4 P.AAX:   .LONG    36                                         ;
                                     00000000'  001B8          .ADDRESS P.AAY                                      ;


                                                               .PSECT   $CODE$,NOWRT,2

                                          007C  00000          .ENTRY   NML$FILEDMP, Save R2,R3,R4,R5,R6           ; 1025
                      56 00000000G  00  9E  00002          MOVAB    LIB$PUT_OUTPUT, R6                         ;
                      55 00000000V  00  9E  00009          MOVAB    NML$APPENDTXT, R5                          ;
                      54 00000000G  00  9E  00010          MOVAB    NML$GL_LOGMASK, R4                         ;
                      53 00000000'  00  9E  00017          MOVAB    P.AAT, R3                                  ;
                      5E      FD5C  CE  9E  0001E          MOVAB    -676(SP), SP                               ;
          73          64        04  AC  E1  00023          BBC      BITNUM, NML$GL_LOGMASK, 3$                 ; 1069
               0C     AE      0100  8F  B0  00028          MOVW     #256, OUTDSC                               ; 1075
               10     AE      FF00  CD  9E  0002E          MOVAB    FAOBUF, OUTDSC+4                           ; 1076
                                0C  AE  9F  00034          PUSHAB   OUTDSC                                     ; 1082
                                10  AE  9F  00037          PUSHAB   OUTDSC                                     ; 1078
                                    53  DD  0003A          PUSHL    R3                                         ; 1080
                                D0  A3  9F  0003C          PUSHAB   P.AAR                                      ; 1079
                                08  AC  DD  0003F          PUSHL    FID                                        ; 1082
                      65            05  FB  00042          CALLS    #5, NML$APPENDTXT                          ;
                                0C  AE  9F  00045          PUSHAB   OUTDSC                                     ; 1084
                      66            01  FB  00048          CALLS    #1, LIB$PUT_OUTPUT                         ;
     52         64    01            01  EF  0004B          EXTZV    #1, #1, NML$GL_LOGMASK, SAVEFLAG           ; 1089
                      64            02  88  00050          BISB2    #2, NML$GL_LOGMASK                         ; 1090
                                    6E  B4  00053          CLRW     KEY                                        ; 1094
                                04  AE  9F  00055 1$:      PUSHAB   RECDSC                                     ; 1095
                         00000000G  00  9F  00058          PUSHAB   NML$GQ_RECBFDSC                            ;
                                08  AE  9F  0005E          PUSHAB   KEY                                        ;
                                08  AC  DD  00061          PUSHL    FID                                        ;
            00000000G 00            04  FB  00064          CALLS    #4, NMA$READREC                            ;
                      04            50  E9  0006B          BLBC     R0, 2$                                     ;
                                    6E  B6  0006E          INCW     KEY                                        ; 1098
                                    E3  11  00070          BRB      1$                                         ; 1095
               0C     AE      0100  8F  B0  00072 2$:      MOVW     #256, OUTDSC                               ; 1104
               10     AE      FF00  CD  9E  00078          MOVAB    FAOBUF, OUTDSC+4                           ; 1105
                                0C  AE  9F  0007E          PUSHAB   OUTDSC                                     ; 1111
                                10  AE  9F  00081          PUSHAB   OUTDSC                                     ; 1107
                                58  A3  9F  00084          PUSHAB   P.AAX                                      ; 1109
                                2C  A3  9F  00087          PUSHAB   P.AAV                                      ; 1108
                                08  AC  DD  0008A          PUSHL    FID                                        ; 1111
                      65            05  FB  0008D          CALLS    #5, NML$APPENDTXT                          ;
                                0C  AE  9F  00090          PUSHAB   OUTDSC                                     ; 1113
                      66            01  FB  00093          CALLS    #1, LIB$PUT_OUTPUT                         ;
     64         01    01            52  F0  00096          INSV     SAVEFLAG, #1, #1, NML$GL_LOGMASK           ; 1117
                                        04  0009B 3$:      RET                                                 ; 1119

; Routine Size:  156 bytes,    Routine Base:  $CODE$ + 0535
```

```
;  1134          1120  1
;  1135          1121  1 %SBTTL 'NML$LOGFILEOP  Log a file operation'
;  1136          1122  1 GLOBAL ROUTINE NML$LOGFILEOP (BITNUM, FILEID, TXTDSC) : NOVALUE =
;  1137          1123  1
;  1138          1124  1 !++
;  1139          1125  1 ! FUNCTIONAL DESCRIPTION:
;  1140          1126  1 !
;  1141          1127  1 !       This routine logs file operations such as open and close.
;  1142          1128  1 !
;  1143          1129  1 ! FORMAL PARAMETERS:
;  1144          1130  1 !
;  1145          1131  1 !       BITNUM          Logging flag bit number.
;  1146          1132  1 !       FILEID          Value of the fileid parameter (NMA$C_OPN_xxxx)
;  1147          1133  1 !       TXTDSC          Descriptor of message text.
;  1148          1134  1 !
;  1149          1135  1 ! IMPLICIT INPUTS:
;  1150          1136  1 !
;  1151          1137  1 !       NONE
;  1152          1138  1 !
;  1153          1139  1 ! IMPLICIT OUTPUTS:
;  1154          1140  1 !
;  1155          1141  1 !       NONE
;  1156          1142  1 !
;  1157          1143  1 ! ROUTINE VALUE:
;  1158          1144  1 ! COMPLETION CODES:
;  1159          1145  1 !
;  1160          1146  1 !       NONE
;  1161          1147  1 !
;  1162          1148  1 ! SIDE EFFECTS:
;  1163          1149  1 !
;  1164          1150  1 !       NONE
;  1165          1151  1 !
;  1166          1152  1 !--
;  1167          1153  1
;  1168          1154  2     BEGIN
;  1169          1155  2
;  1170          1156  2     LOCAL
;  1171          1157  2         OUTBUF : VECTOR [255, BYTE],
;  1172          1158  2         OUTDSC : DESCRIPTOR;
;  1173          1159  2
;  1174          1160  2     OUTDSC [DSC$W_LENGTH] = 255;
;  1175          1161  2     OUTDSC [DSC$A_POINTER] = OUTBUF;
;  1176          1162  2
;  1177          1163  2     NML$APPENDTXT (.FILEID, 0, .TXTDSC, OUTDSC, OUTDSC [DSC$W_LENGTH]);
;  1178          1164  2
;  1179          1165  2     NML$DEBUG_TXT (.BITNUM,
;  1180          1166  2                    OUTDSC);
;  1181          1167  1     END;


                                        0000 00000          .ENTRY  NML$LOGFILEOP, Save nothing                 ; 1122
                              5E    FEF8   CE   9E 00002    MOVAB   -264(SP), SP                                ;
                              6E      FF   8F   9B 00007    MOVZBW  #255, OUTDSC                                ; 1160
                        04    AE      08   AE   9E 0000B    MOVAB   OUTBUF, OUTDSC+4                            ; 1161
```

```
                                    5E   DD 00010          PUSHL   SP                                  ; 1163
                                04  AE   9F 00012          PUSHAB  OUTDSC                              ;
                                0C  AC   DD 00015          PUSHL   TXTDSC                              ;
                                    7E   D4 00018          CLRL    -(SP)                               ;
                                08  AC   DD 0001A          PUSHL   FILEID                              ;
                  00000000V     00       05 FB 0001D      CALLS   #5, NML$APPENDTXT                   ;
                                    5E   DD 00024          PUSHL   SP                                  ; 1165
                                04  AC   DD 00026          PUSHL   BITNUM                              ;
                      FC2E      CF       02 FB 00029      CALLS   #2, NML$DEBUG_TXT                   ;
                                         04 0002E          RET                                         ; 1167

; Routine Size:  47 bytes,    Routine Base:  $CODE$ + 05D1
```

```
;  1183          1168  1 %SBTTL 'NML$LOGRECORDOP  Log a record access operation'
;  1184          1169  1 GLOBAL ROUTINE NML$LOGRECORDOP (BITNUM, FILEID, TXTDSC, DATDSC) : NOVALUE =
;  1185          1170  1
;  1186          1171  1 !++
;  1187          1172  1 ! FUNCTIONAL DESCRIPTION:
;  1188          1173  1 !
;  1189          1174  1 !       This routine logs record access operations such as read, write,
;  1190          1175  1 !       and delete.
;  1191          1176  1 !
;  1192          1177  1 ! FORMAL PARAMETERS:
;  1193          1178  1 !
;  1194          1179  1 !       BITNUM          Logging flag bit number.
;  1195          1180  1 !       FILEID          Value of the fileid parameter (NMA$C_OPN_xxxx).
;  1196          1181  1 !       TXTDSC          Descriptor of message text.
;  1197          1182  1 !       DATDSC          Descriptor of record data.
;  1198          1183  1 !
;  1199          1184  1 ! IMPLICIT INPUTS:
;  1200          1185  1 !
;  1201          1186  1 !       NONE
;  1202          1187  1 !
;  1203          1188  1 ! IMPLICIT OUTPUTS:
;  1204          1189  1 !
;  1205          1190  1 !       NONE
;  1206          1191  1 !
;  1207          1192  1 ! ROUTINE VALUE:
;  1208          1193  1 ! COMPLETION CODES:
;  1209          1194  1 !
;  1210          1195  1 !       NONE
;  1211          1196  1 !
;  1212          1197  1 ! SIDE EFFECTS:
;  1213          1198  1 !
;  1214          1199  1 !       NONE
;  1215          1200  1 !
;  1216          1201  1 !--
;  1217          1202  1
;  1218          1203  2     BEGIN
;  1219          1204  2
;  1220          1205  2     MAP
;  1221          1206  2         DATDSC : REF DESCRIPTOR;
;  1222          1207  2
;  1223          1208  2     LOCAL
;  1224          1209  2         OUTBUF : VECTOR [255, BYTE],
;  1225          1210  2         OUTDSC : DESCRIPTOR;
;  1226          1211  2 !
;  1227          1212  2 ! Initialize output buffer descriptor.
;  1228          1213  2 !
;  1229          1214  2     OUTDSC [DSC$W_LENGTH] = 255;
;  1230          1215  2     OUTDSC [DSC$A_POINTER] = OUTBUF;
;  1231          1216  2 !
;  1232          1217  2 ! Append the file type to the message text.
;  1233          1218  2 !
;  1234          1219  2     NML$APPENDTXT (.FILEID, 0, .TXTDSC, OUTDSC, OUTDSC [DSC$W_LENGTH]);
;  1235          1220  2 !
;  1236          1221  2 ! Log the data.
;  1237          1222  2 !
;  1238          1223  2     NML$DEBUG_MSG (.BITNUM,
;  1239          1224  2                    .DATDSC [DSC$A_POINTER],
```

```
;  1240          1225  2                     .DATDSC [DSC$W_LENGTH],
;  1241          1226  2                     OUTDSC);
;  1242          1227  2
;  1243          1228  1     END;


                                                0000 00000          .ENTRY  NML$LOGRECORDOP, Save nothing                ; 1169
                                      5E     FEF8   CE   9E 00002   MOVAB   -264(SP), SP                                 ;
                                      6E       FF   8F   9B 00007   MOVZBW  #255, OUTDSC                                 ; 1214
                                   04 AE       08   AE   9E 0000B   MOVAB   OUTBUF, OUTDSC+4                            ; 1215
                                                    5E   DD 00010   PUSHL   SP                                         ; 1219
                                                04  AE   9F 00012   PUSHAB  OUTDSC                                      ;
                                                0C  AC   DD 00015   PUSHL   TXTDSC                                     ;
                                                    7E   D4 00018   CLRL    -(SP)                                      ;
                                                08  AC   DD 0001A   PUSHL   FILEID                                      ;
                                00000000V       00       05   FB 0001D   CALLS   #5, NML$APPENDTXT                      ;
                                                    5E   DD 00024   PUSHL   SP                                         ; 1223
                                      50        10   AC   D0 00026   MOVL    DATDSC, R0                                 ; 1225
                                      7E             60   3C 0002A   MOVZWL  (R0), -(SP)                               ;
                                                04  A0   DD 0002D   PUSHL   4(R0)                                       ; 1224
                                                04  AC   DD 00030   PUSHL   BITNUM                                     ; 1223
                                    FC33        CF       04   FB 00033   CALLS   #4, NML$DEBUG_MSG                     ;
                                                         04   00038   RET                                             ; 1228

; Routine Size:  57 bytes,    Routine Base:  $CODE$ + 0600
```

```
;  1245          1229  1 %SBTTL 'NML$APPENDTXT  Append file type to string'
;  1246          1230  1 GLOBAL ROUTINE NML$APPENDTXT (FILEID, PFXDSC, SFXDSC, OUTDSC, RESLEN) : NOVALUE =
;  1247          1231  1
;  1248          1232  1 !++
;  1249          1233  1 ! FUNCTIONAL DESCRIPTION:
;  1250          1234  1 !
;  1251          1235  1 !       This routine appends the file type string to the beginning of a
;  1252          1236  1 !       text string.
;  1253          1237  1 !
;  1254          1238  1 ! FORMAL PARAMETERS:
;  1255          1239  1 !
;  1256          1240  1 !       FILEID          Value of the fileid parameter (NMA$C_OPN_xxxx)
;  1257          1241  1 !       PFXDSC          Descriptor of prefix text.
;  1258          1242  1 !       SFXDSC          Descriptor of suffix text.
;  1259          1243  1 !       OUTDSC          Descriptor of output string buffer.
;  1260          1244  1 !       RESLEN          Length of resulting appended string.
;  1261          1245  1 !
;  1262          1246  1 ! IMPLICIT INPUTS:
;  1263          1247  1 !
;  1264          1248  1 !       NONE
;  1265          1249  1 !
;  1266          1250  1 ! IMPLICIT OUTPUTS:
;  1267          1251  1 !
;  1268          1252  1 !       NONE
;  1269          1253  1 !
;  1270          1254  1 ! ROUTINE VALUE:
;  1271          1255  1 ! COMPLETION CODES:
;  1272          1256  1 !
;  1273          1257  1 !       NONE
;  1274          1258  1 !
;  1275          1259  1 ! SIDE EFFECTS:
;  1276          1260  1 !
;  1277          1261  1 !       NONE
;  1278          1262  1 !
;  1279          1263  1 !--
;  1280          1264  1
;  1281          1265  2     BEGIN
;  1282          1266  2
;  1283          1267  2     MAP
;  1284          1268  2         PFXDSC : REF DESCRIPTOR,
;  1285          1269  2         SFXDSC : REF DESCRIPTOR,
;  1286          1270  2         OUTDSC : REF DESCRIPTOR;
;  1287          1271  2
;  1288          1272  2     LOCAL
;  1289          1273  2         CTLDSC : REF DESCRIPTOR,
;  1290          1274  2         FAOLST : VECTOR [6],
;  1291          1275  2         IDX,
;  1292          1276  2         TYPDSC : REF DESCRIPTOR;
;  1293          1277  2 !
;  1294          1278  2 ! Select file type string.
;  1295          1279  2 !
;  1296          1280  2     CASE .FILEID FROM NMA$C_OPN_MIN TO NMA$C_OPN_MAX OF
;  1297          1281  2         SET
;  1298          1282  2         [NMA$C_OPN_NODE]:
;  1299          1283  2             TYPDSC = $ASCID ('Node');
;  1300          1284  2         [NMA$C_OPN_LINE]:
;  1301          1285  2             TYPDSC = $ASCID ('Line');
```

```
;  1302     1286  2          [NMA$C_OPN_LOG]:
;  1303     1287  2              TYPDSC = $ASCID ('Logging');
;  1304     1288  2          [NMA$C_OPN_OBJ]:
;  1305     1289  2              TYPDSC = $ASCID ('Object');
;  1306     1290  2          [NMA$C_OPN_CIR]:
;  1307     1291  2              TYPDSC = $ASCID ('Circuit');
;  1308     1292  2          [NMA$C_OPN_X25]:
;  1309     1293  2              TYPDSC = $ASCID ('X25 Module');
;  1310     1294  2          [NMA$C_OPN_X29]:
;  1311     1295  2              TYPDSC = $ASCID ('X29 Module');
;  1312     1296  2          [NMA$C_OPN_CNF]:
;  1313     1297  2              TYPDSC = $ASCID ('Configurator Module');
;  1314     1298  2          [INRANGE,
;  1315     1299  2           OUTRANGE]:
;  1316     1300  2              TYPDSC = $ASCID ('Unknown');
;  1317     1301  2          TES;
;  1318     1302  2
;  1319     1303  2      ! Set up FAO parameters.
;  1320     1304  2      !
;  1321     1305  2          IDX = 0;
;  1322     1306  2
;  1323     1307  2          IF .PFXDSC NEQA 0
;  1324     1308  2          THEN
;  1325     1309  3              BEGIN
;  1326     1310  3
;  1327     1311  3              FAOLST [0] = .PFXDSC [DSC$W_LENGTH];
;  1328     1312  3              FAOLST [1] = .PFXDSC [DSC$A_POINTER];
;  1329     1313  3              IDX = 2;
;  1330     1314  3
;  1331     1315  2              END;
;  1332     1316  2
;  1333     1317  2          FAOLST [.IDX] = .TYPDSC [DSC$W_LENGTH];
;  1334     1318  2          FAOLST [.IDX + 1] = .TYPDSC [DSC$A_POINTER];
;  1335     1319  2          IDX = .IDX + 2;
;  1336     1320  2
;  1337     1321  2          IF .SFXDSC NEQA 0
;  1338     1322  2          THEN
;  1339     1323  3              BEGIN
;  1340     1324  3
;  1341     1325  3              FAOLST [.IDX] = .SFXDSC [DSC$W_LENGTH];
;  1342     1326  3              FAOLST [.IDX + 1]   .SFXDSC [DSC$A_POINTER];
;  1343     1327  3              IDX = .IDX + 2;
;  1344     1328  3
;  1345     1329  2              END;
;  1346     1330  2
;  1347     1331  2          IF .IDX GTR 4
;  1348     1332  2          THEN
;  1349     1333  3              CTLDSC = $ASCID ('!AD !AD !AD')
;  1350     1334  2          ELSE
;  1351     1335  2              CTLDSC = $ASCID ('!AD !AD');
;  1352     1336  2      !
;  1353     1337  2      ! Append the name to the beginning of the text string.
;  1354     1338  2      !
;  1355   P 1339  2          $FAOL (CTRSTR = .CTLDSC,
;  1356   P 1340  2                 OUTLEN = .RESLEN,
;  1357   P 1341  2                 OUTBUF = .OUTDSC,
;  1358     1342  2                 PRMLST = FAOLST);
```

```
;1359            1343  2
;1360            1344  1     END;


                                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

                                                    65  64  6F  4E  001BC P.ABA:  .ASCII  \Node\                                  ;
                                                      00000004  001C0 P.AAZ:  .LONG   4                                           ;
                                                      00000000' 001C4          .ADDRESS P.ABA                                     ;
                                                    65  6E  69  4C  001C8 P.ABC:  .ASCII  \Line\                                  ;
                                                      00000004  001CC P.ABB:  .LONG   4                                           ;
                                                      00000000' 001D0          .ADDRESS P.ABC                                     ;
                                            67  6E  69  67  67  6F  4C  001D4 P.ABE:  .ASCII  \Logging\                           ;
                                                                001DB        .BLKB   1                                           ;
                                                      00000007  001DC P.ABD:  .LONG   7                                           ;
                                                      00000000' 001E0          .ADDRESS P.ABE                                     ;
                                                74  63  65  6A  62  4F  001E4 P.ABG:  .ASCII  \Object\                            ;
                                                                001EA        .BLKB   2                                           ;
                                                      00000006  001EC P.ABF:  .LONG   6                                           ;
                                                      00000000' 001F0          .ADDRESS P.ABG                                     ;
                                            74  69  75  63  72  69  43  001F4 P.ABI:  .ASCII  \Circuit\                           ;
                                                                001FB        .BLKB   1                                           ;
                                                      00000007  001FC P.ABH:  .LONG   7                                           ;
                                                      00000000' 00200          .ADDRESS P.ABI                                     ;
                                65  6C  75  64  6F  4D  20  35  32  58  00204 P.ABK:  .ASCII  \X25 Module\                        ;
                                                                0020E        .BLKB   2                                           ;
                                                      0000000A  00210 P.ABJ:  .LON    10                                          ;
                                                      00000000' 00214          .ADDRESS P.ABK                                     ;
                                65  6C  75  64  6F  4D  20  39  32  58  00218 P.ABM:  .ASCII  \X29 Module\                        ;
                                                                00222        .BLKB   2                                           ;
                                                      0000000A  00224 P.ABL:  .LONG   10                                          ;
                                                      00000000' 00228          .ADDRESS P.ABM                                     ;
6F  4D  20  72  6F  74  61  72  75  67  69  66  6E  6F  43  0022C P.ABO:  .ASCII  \Configurator Module\                            ;
                                                        65  6C  75  64  0023B                                                     ;
                                                                0023F        .BLKB   1                                           ;
                                                      00000013  00240 P.ABN:  .LONG   19                                          ;
                                                      00000000' 00244          .ADDRESS P.ABO                                     ;
                                            6E  77  6F  6E  6B  6E  55  00248 P.ABQ:  .ASCII  \Unknown\                           ;
                                                                0024F        .BLKB   1                                           ;
                                                      00000007  00250 P.ABP:  .LONG   7                                           ;
                                                      00000000' 00254          .ADDRESS P.ABQ                                     ;
                                44  41  21  20  44  41  21  20  44  41  21  00258 P.ABS:  .ASCII  \!AD !AD !AD\                    ;
                                                                00263        .BLKB   1                                           ;
                                                      0000000B  00264 P.ABR:  .LONG   11                                          ;
                                                      00000000' 00268          .ADDRESS P.ABS                                     ;
                                            44  41  21  20  44  41  21  0026C P.ABU:  .ASCII  \!AD !AD\                           ;
                                                                00273        .BLKB   1                                           ;
                                                      00000007  00274 P.ABT:  .LONG   7                                           ;
                                                      00000000' 00278          .ADDRESS P.ABU                                     ;


                                                                                .PSECT  $CODE$,NOWRT,2

                                                          000C 00000           .ENTRY  NML$APPENDTXT, Save R2,R3                  ; 1230
                                        53 00000000'  00  9E 00002             MOVAB   P.ABP, R3                                  ;
```

```
                                  5E          18  C2 00009          SUBL2   #24, SP                             ;
                        07        00      04  AC  CF 0000C          CASEL   FILEID, #0, #7                      ; 1280
        0029          0023      001C        0015     00011 1$:      .WORD   2$-1$,-                             ;
        0041          003B      0035        002F     00019                  3$-1$,-                             ;
                                                                            4$-1$,-                             ;
                                                                            5$-1$,-                             ;
                                                                            6$-1$,-                             ;
                                                                            7$-1$,-                             ;
                                                                            8$-1$,-                             ;
                                                                            9$-1$                               ;
                                  52          63  9E 00021          MOVAB   P.ABP, TYPDSC                       ; 1300
                                              30  11 00024          BRB     10$                                 ;
                                  52    FF70  C3  9E 00026 2$:      MOVAB   P.AAZ, TYPDSC                       ; 1283
                                              29  11 0002B          BRB     10$                                 ;
                                  52    FF7C  C3  9E 0002D 3$:      MOVAB   P.ABB, TYPDSC                       ; 1285
                                              22  11 00032          BRB     10$                                 ;
                                  52      8C  A3  9E 00034 4$:      MOVAB   P.ABD, TYPDSC                       ; 1287
                                              1C  11 00038          BRB     10$                                 ;
                                  52      9C  A3  9E 0003A 5$:      MOVAB   P.ABF, TYPDSC                       ; 1289
                                              16  11 0003E          BRB     10$                                 ;
                                  52      AC  A3  9E 00040 6$:      MOVAB   P.ABH, TYPDSC                       ; 1291
                                              10  11 00044          BRB     10$                                 ;
                                  52      C0  A3  9E 00046 7$:      MOVAB   P.ABJ, TYPDSC                       ; 1293
                                              0A  11 0004A          BRB     10$                                 ;
                                  52      D4  A3  9E 0004C 8$:      MOVAB   P.ABL, TYPDSC                       ; 1295
                                              04  11 00050          BRB     10$                                 ;
                                  52      F0  A3  9E 00052 9$:      MOVAB   P.ABN, TYPDSC                       ; 1297
                                              50  D4 00056 10$:     CLRL    IDX                                 ; 1305
                                  51      08  AC  D0 00058          MOVL    PFXDSC, R1                          ; 1307
                                              0B  13 0005C          BEQL    11$                                 ;
                                  6E          61  3C 0005E          MOVZWL  (R1), FAOLST                        ; 1311
                             04   AE      04  A1  D0 00061          MOVL    4(R1), FAOLST+4                     ; 1312
                                  50          02  D0 00066          MOVL    #2, IDX                             ; 1313
                                6E40          62  3C 00069 11$:     MOVZWL  (TYPDSC), FAOLST[IDX]               ; 1317
                             04 AE40      04  A2  D0 0006D          MOVL    4(TYPDSC), FAOLST+4[IDX]            ; 1318
                                  50          02  C0 00073          ADDL2   #2, IDX                             ; 1319
                                  51      0C  AC  D0 00076          MOVL    SFXDSC, R1                          ; 1321
                                              0D  13 0007A          BEQL    12$                                 ;
                                6E40          61  3C 0007C          MOVZWL  (R1), FAOLST[IDX]                   ; 1325
                             04 AE40      04  A1  D0 00080          MOVL    4(R1), FAOLST+4[IDX]                ; 1326
                                  50          02  C0 00086          ADDL2   #2, IDX                             ; 1327
                                  04          50  D1 00089 12$:     CMPL    IDX, #4                             ; 1331
                                              06  15 0008C          BLEQ    13$                                 ;
                                  50      14  A3  9E 0008E          MOVAB   P.ABR, CTLDSC                       ; 1333
                                              04  11 00092          BRB     14$                                 ;
                                  50      24  A3  9E 00094 13$:     MOVAB   P.ABT, CTLDSC                       ; 1335
                                              5E  DD 00098 14$:     PUSHL   SP                                  ; 1342
                                          10  AC  DD 0009A          PUSHL   OUTDSC                              ;
                                          14  AC  DD 0009D          PUSHL   RESLEN                              ;
                                              50  DD 000A0          PUSHL   CTLDSC                              ;
                        00000000G 00          04  FB 000A2          CALLS   #4, SYS$FAOL                        ;
                                                  04 000A9          RET                                         ; 1344

; Routine Size:  170 bytes,    Routine Base:  $CODE$ + 0639
```

```
;  1362          1345  1 %SBTTL 'NML$TRNLOGNUM  Translate numeric logical name'
;  1363          1346  1 GLOBAL ROUTINE NML$TRNLOGNUM (LNMDSC, RESADR) =
;  1364          1347  1
;  1365          1348  1 !++
;  1366          1349  1 ! FUNCTIONAL DESCRIPTION:
;  1367          1350  1 !
;  1368          1351  1 !       This routine translates a logical name and returns the numeric
;  1369          1352  1 !       representation of the ASCII hexadecimal number that results.
;  1370          1353  1 !
;  1371          1354  1 ! FORMAL PARAMETERS:
;  1372          1355  1 !
;  1373          1356  1 !       LNMDSC          Descriptor of the logical name to be translated.
;  1374          1357  1 !       RESADR          Address of longword to contain the numeric value.
;  1375          1358  1 !
;  1376          1359  1 ! IMPLICIT INPUTS:
;  1377          1360  1 !
;  1378          1361  1 !       NONE
;  1379          1362  1 !
;  1380          1363  1 ! IMPLICIT OUTPUTS:
;  1381          1364  1 !
;  1382          1365  1 !       NONE
;  1383          1366  1 !
;  1384          1367  1 ! ROUTINE VALUE:
;  1385          1368  1 ! COMPLETION CODES:
;  1386          1369  1 !
;  1387          1370  1 !       Returns error code if the logical name has no translation or the
;  1388          1371  1 !       translation is invalid.  The result longword will be set to zero.
;  1389          1372  1 !
;  1390          1373  1 ! SIDE EFFECTS:
;  1391          1374  1 !
;  1392          1375  1 !       NONE
;  1393          1376  1 !
;  1394          1377  1 !--
;  1395          1378  1
;  1396          1379  2     BEGIN
;  1397          1380  2
;  1398          1381  2     MAP
;  1399          1382  2         LNMDSC : DESCRIPTOR;
;  1400          1383  2
;  1401          1384  2     OWN
;  1402          1385  2         ASCLEN : WORD,
;  1403          1386  2         ASCNUM : VECTOR [8, BYTE];
;  1404          1387  2
;  1405          1388  2     LOCAL
;  1406          1389  2         STATUS;
;  1407          1390  2
;  1408        P 1391  2     STATUS = $TRNLNM (ATTR = UPLIT (LNM$M_CASE_BLIND),
;  1409        P 1392  2                      TABNAM = %ASCID 'LNM$PROCESS_TABLE',
;  1410        P 1393  2                      LOGNAM = .LNMDSC,
;  1411        P 1394  2                      ITMLST = UPLIT (WORD (8), WORD (LNM$_STRING),
;  1412        P 1395  2                                      LONG (ASCNUM),
;  1413        P 1396  2                                      LONG (ASCLEN),
;  1414        P 1397  2                                      LONG (0))
;  1415          1398  2                      );
;  1416          1399  2
;  1417          1400  2     IF .STATUS EQL SS$_NORMAL
;  1418          1401  2     THEN
```

```
; 1419          1402  2          STATUS = LIB$CVT_HTB (.ASCLEN, ASCNUM, .RESADR);
; 1420          1403  2
; 1421          1404  2      RETURN .STATUS
; 1422          1405  2
; 1423          1406  1      END;                                        . End of NML$TRNLOGNUM


                                                                           .PSECT  $PLIT$,NOWRT,NOEXE,2

                                             02000000  0027C P.ABV:  .LONG   33554432
42 41 54 5F 53 53 45 43 4F 52 50 24 4D 4E 4C  00280 P.ABX:  .ASCII  \LNM$PROCESS_TABLE\<0><0><0>
                              00 00 00 45 4C  0028F
                                             010E0011  00294 P.ABW:  .LONG   17694737
                                            00000000'  00298         .ADDRESS P.ABX
                                                 0008  0029C P.ABY:  .WORD   8
                                                 0002  0029E         .WORD   2
                                            00000000'  002A0         .ADDRESS ASCNUM
                                            00000000'  002A4         .ADDRESS ASCLEN
                                             00000000  002A8         .LONG   0

                                                                           .PSECT  $OWN$,NOEXE,2

                                                       00120 ASCLEN: .BLKB   2
                                                       00122         .BLKB   2
                                                       00124 ASCNUM: .BLKB   8

                                                                           .EXTRN  SYS$TRNLNM

                                                                           .PSECT  $CODE$,NOWRT,2

                                                 0004  00000         .ENTRY  NML$TRNLOGNUM, Save R2            ; 1346
                           52 00000000'   00   9E  00002         MOVAB   P.ABY, R2                        ;
                                          52   DD  00009         PUSHL   R2                               ; 1398
                                          7E   D4  0000B         CLRL    -(SP)                            ;
                                     04   AC   DD  0000D         PUSHL   LNMDSC                           ;
                                     F8   A2   9F  00010         PUSHAB  P.ABW                            ;
                                     E0   A2   9F  00013         PUSHAB  P.ABV                            ;
                     00000000G  00        05   FB  00016         CALLS   #5, SYS$TRNLNM                   ;
                                01        50   D1  0001D         CMPL    STATUS, #1                       ; 1400
                                          17   12  00020         BNEQ    1$                               ;
                                     08   AC   DD  00022         PUSHL   RESADR                           ; 1402
                              00000000'   00   9F  00025         PUSHAB  ASCNUM                           ;
                           7E 00000000'   00   3C  0002B         MOVZWL  ASCLEN, -(SP)                    ;
                     00000000G  00        03   FB  00032         CALLS   #3, LIB$CVT_HTB                  ;
                                          04       00039 1$:     RET                                      ; 1406

; Routine Size:  58 bytes,    Routine Base:  $CODE$ + 06E3


; 1424          1407  1
```

```
; 1426          1408  1 END                                       ! End of module
; 1427          1409  1
; 1428          1410  0 ELUDOM



                                                                .EXTRN  LIB$SIGNAL

;                                   PSECT SUMMARY

;       Name                    Bytes                           Attributes

;   $CODE$                        1821  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;   $OWN$                          300  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;   $PLIT$                         684  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)



;                               Library Statistics

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

;   _$255$DUA28:[NML.OBJ]NMLLIB.L32;1     341        48        14          27        00:00.1
;   _$255$DUA28:[SHRLIB]NMALIBRY.L32;1    887        15         1          47        00:00.2
;   _$255$DUA28:[SYSLIB]STARLET.L32;1    9776        16         0         581        00:02.1



;                                       COMMAND QUALIFIERS

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

; 1429          1411  0
; Size:         1821 code + 984 data bytes
; Run Time:       00:37.4
; Elapsed Time:   01:27.0
; Lines/CPU Min:   2261
; Lexemes/CPU-Min: 19199
; Memory Used:  157 pages
; Compilation Complete
```

0281 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

NMLCLEAR
LIS

NMLBLDMSG
LIS

NMLCHANGE
LIS

NMLDAT
LIS

NMLCLPUST
LIS