```
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**NMLPARPRM

```
NN       NN  MM       MM  LL          PPPPPPPP      AAAAAA    RRRRRRRR    PPPPPPPP    RRRRRRRR    MM       MM
NN       NN  MM       MM  LL          PPPPPPPP      AAAAAA    RRRRRRRR    PPPPPPPP    RRRRRRRR    MM       MM
NN       NN  MMMM   MMMM  LL          PP      PP  AA      AA  RR      RR  PP      PP  RR      RR  MMMM   MMMM
NN       NN  MMMM   MMMM  LL          PP      PP  AA      AA  RR      RR  PP      PP  RR      RR  MMMM   MMMM
NNNN     NN  MM  MM  MM  LL          PP      PP  AA      AA  RR      RR  PP      PP  RR      RR  MM  MM  MM
NNNN     NN  MM  MM  MM  LL          PP      PP  AA      AA  RR      RR  PP      PP  RR      RR  MM  MM  MM
NN  NN   NN  MM      MM  LL          PPPPPPPP    AA      AA  RRRRRRRR    PPPPPPPP    RRRRRRRR    MM      MM
NN  NN   NN  MM      MM  LL          PPPPPPPP    AA      AA  RRRRRRRR    PPPPPPPP    RRRRRRRR    MM      MM
NN    NNNN  MM      MM  LL          PP          AAAAAAAAAA  RR  RR      PP          RR  RR      MM      MM
NN    NNNN  MM      MM  LL          PP          AAAAAAAAAA  RR  RR      PP          RR  RR      MM      MM
NN      NN  MM      MM  LL          PP          AA      AA  RR    RR    PP          RR    RR    MM      MM      ....
NN      NN  MM      MM  LL          PP          AA      AA  RR    RR    PP          RR    RR    MM      MM      ....
NN      NN  MM      MM  LLLLLLLLLL  PP          AA      AA  RR      RR  PP          RR      RR  MM      MM      ....
NN      NN  MM      MM  LLLLLLLLLL  PP          AA      AA  RR      RR  PP          RR      RR  MM      MM      ....

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 NPARSE action routines for parsing parameters'
;    2          0002  0 MODULE NML$PARPRM (
;    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 V2.0 Network Management Listener
;   36          0036  1 !
;   37          0037  1 !
;   38          0038  1 ! ABSTRACT:
;   39          0039  1 !
;   40          0040  1 !    This module contains action routines called by NPARSE to parse and
;   41          0041  1 !    store NICE ertity parameters.
;   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:  8-Nov-1979
;   48          0048  1 !
;   49          0049  1 ! MODIFIED BY:
;   50          0050  1 !
;   51          0051  1 !       V03-007 MKP0007         Kathy Perko             25-Mar-1984
;   52          0052  1 !               Add support to convert area 0 to 1 for Phase IV NCPs
;   53          0053  1 !               and to the executor's area for Phase III NCPs.
;   54          0054  1 !
;   55          0055  1 !       V03-006 MKP0006         Kathy Perko             9-Jan-1984
;   56          0056  1 !               Change name of NML$PRM_CHKHOST to the more general
;   57          0057  1 !               name, NML$PRM_CHKNODE.  Modify NML$PRM_CIRC_OWNER to
```

```
;    58          0058  1 !                 check the node to see if the executor.
;    59          0059  1 !
;    60          0060  1 !         V03-005 MKP0005         Kathy Perko             4-Aug-1983
;    61          0061  1 !                 Add support to make node permanent database faster.
;    62          0062  1 !
;    63          0063  1 !         V03-004 MKP0004         Kathy Perko             20-April-1983
;    64          0064  1 !                 Remove service functions from NML.
;    65          0065  1 !
;    66          0066  1 !         V03-003 MKP0003         Kathy Perko             31-Aug-1982
;    67          0067  1 !                 Fix X25-PROTOCOL GROUP checks to make sure right
;    68          0068  1 !                 combination of groups and dtes are in command.
;    69          0069  1 !
;    70          0070  1 !         V03-002 MKP0002         Kathy Perko             21-June-1982
;    71          0071  1 !                 Add routine to use NET string as entity ID if NICE
;    72          0072  1 !                 command is SET X-PROTOCOL NET <net-id>.  Otherwise,
;    73          0073  1 !                 NML uses a null string to indicate the active network.
;    74          0074  1 !                 Also, add a routine to make sure KNOWN qualifiers are
;    75          0075  1 !                 used only with ALL (for X25-PROTOCOL GROUPS).
;    76          0076  1 !
;    77          0077  1 !         V03-001 MKP0001         Kathy Perko             16-June-1982
;    78          0078  1 !                 Add routine to parse the circuit parameter, owner.
;    79          0079  1 !
;    80          0080  1 !         V02-004 MKP0003         Kathy Perko             23-Feb-1982
;    81          0081  1 !                 X25-PROTOCOL GROUP commands with DTE IDs included.  This
;    82          0082  1 !                 routine concatenates a root entity with a sub entity
;    83          0083  1 !                 to form the entity id.  Used when the ACPs data base keeps
;    84          0084  1 !                 the entities in a hierarchical form.  Also, delete the routine,
;    85          0085  1 !                 NML$PRM_CHK_PROTOCOL.
;    86          0086  1 !
;    87          0087  1 !         V02-003 MKP0002         Kathy Perko             15-Dec-1981
;    88          0088  1 !                 Change name NML$PRS_CHK_PROTOCOL to NML$PRM_CHK_PROTOCOL
;    89          0089  1 !                 to be consistent with routine names in this module.
;    90          0090  1 !
;    91          0091  1 !         V02-002 MKP0001         Kathy Perko             19-Nov-1981
;    92          0092  1 !                 Add parameter grouping check for X.25 Protocol module.
;    93          0093  1 !
;    94          0094  1 !         V02-001 LMK0001         Len Kawell              27-Jul-1981
;    95          0095  1 !                 Remove line name parsing, as NETACP handles it now.
;    96          0096  1 !--
;    97          0097  1
```

```
;    99          0098  1 %SBTTL 'Declarations'
;   100          0099  1
;   101          0100  1 !
;   102          0101  1 ! TABLE OF CONTENTS:
;   103          0102  1 !
;   104          0103  1
;   105          0104  1 FORWARD ROUTINE
;   106          0105  1     nml$prm_check,
;   107          0106  1     nml$prm_objprv,
;   108          0107  1     nml$prm_clear,
;   109          0108  1     nml$prm_strchk,
;   110          0109  1     nml$prm_chkexe,
;   111          0110  1     nml$prm_chknod,
;   112          0111  1     nml$prm_chkrem,
;   113          0112  1     nml$prm_chkloo,
;   114          0113  1     nml$prm_chkefi,
;   115          0114  1     nml$prm_chkesi,
;   116          0115  1     nml$prm_circ_owner,
;   117          0116  1     nml$prm_err,
;   118          0117  1     nml$prm_evtsrctyp,
;   119          0118  1     nml$prm_evtsource,
;   120          0119  1     nml$prm_evtclass,
;   121          0120  1     nml$prm_evtmsktyp,
;   122          0121  1     nml$prm_evtmask,
;   123          0122  1     nml$prm_chkeve,
;   124          0123  1     nml$prm_save_node,
;   125          0124  1     nml$prm_set_net,
;   126          0125  1     nml$prm_qual_format,
;   127          0126  1     nml$prm_channels;
;   128          0127  1
;   129          0128  1 !
;   130          0129  1 ! INCLUDE FILES:
;   131          0130  1 !
;   132          0131  1
;   133          0132  1 LIBRARY 'LIB$:NMLLIB.L32',
;   134          0133  1 LIBRARY 'SHRLIB$:NMALIBRY.L32';
;   135          0134  1 LIBRARY 'SYS$LIBRARY:STARLET.L32';
;   136          0135  1
;   137          0136  1 !
;   138          0137  1 ! EXTERNAL REFERENCES:
;   139          0138  1 !
;   140          0139  1
;   141          0140  1 $nml_extdef;
;   142          0141  1
;   143          0142  1 EXTERNAL ROUTINE
;   144          0143  1     nml$chkexe,
;   145          0144  1     nml$chkprmval,
;   146          0145  1     nml$error_2,
;   147          0146  1     nml$fix_node_num,
;   148          0147  1     nml$getnodadr,
;   149          0148  1     nml$saveparam;
;   150          0149  1
```

```
;  152          0150  1 %SBTTL 'NML$PRM_CHECK  Check parameter value'
;  153          0151  1 GLOBAL ROUTINE NML$PRM_CHECK =
;  154          0152  1
;  155          0153  1 !++
;  156          0154  1 ! FUNCTIONAL DESCRIPTION:
;  157          0155  1 !
;  158          0156  1 !       This is a NPARSE action routine that checks a parameter value
;  159          0157  1 !       against its description in the parameter semantic table.  If
;  160          0158  1 !       the parameter is valid, a descriptor entry is created.
;  161          0159  1 !
;  162          0160  1 ! FORMAL PARAMETERS:
;  163          0161  1 !
;  164          0162  1 !       NONE
;  165          0163  1 !
;  166          0164  1 ! IMPLICIT INPUTS:
;  167          0165  1 !
;  168          0166  1 !       NPARSE_BLOCK (pointed to by AP) contains the parsed parameter data.
;  169          0167  1 !           NPA$L_FLDCNT is the parameter length.
;  170          0168  1 !           NPA$L_FLDPTR is a pointer to the parameter in the received
;  171          0169  1 !               message buffer.
;  172          0170  1 !       NML$GL_PRMCODE contains the index into the change parameter table
;  173          0171  1 !           (NML$AB_CPTABLE).
;  174          0172  1 !
;  175          0173  1 ! IMPLICIT OUTPUTS:
;  176          0174  1 !
;  177          0175  1 !       If the parameter is valid then a descriptor entry will be created for
;  178          0176  1 !       it in NML$AW_PRM_DES and NML$GW_PRMDESCNT will be incremented.
;  179          0177  1 !
;  180          0178  1 ! ROUTINE VALUE:
;  181          0179  1 ! COMPLETION CODES:
;  182          0180  1 !
;  183          0181  1 !       Always returns NML$_STS_SUC.
;  184          0182  1 !
;  185          0183  1 ! SIDE EFFECTS:
;  186          0184  1 !
;  187          0185  1 !       If the parameter fails the semantic check, an error message
;  188          0186  1 !       (NMA$C_STS_PVA) is signalled.  If the parameter descriptor buffer
;  189          0187  1 !       is full then a software error (NMA$C_STS_MPR) is signalled with
;  190          0188  1 !       optional text to identify the error.
;  191          0189  1 !
;  192          0190  1 !--
;  193          0191  1
;  194          0192  2     BEGIN
;  195          0193  2
;  196          0194  2     $NPA_ARGDEF;                        ! Define NPARSE block reference
;  197          0195  2
;  198          0196  2     LOCAL
;  199          0197  2         MSGSIZE,                        ! Resultant message size
;  200          0198  2         LEN,
;  201          0199  2         PTR;                            ! Temporary parameter pointer
;  202          0200  2 !
;  203          0201  2 ! Add descriptor entry for this parameter.
;  204          0202  2 !
;  205          0203  2     LEN = .NPARSE_BLOCK [NPA$L_FLDCNT];
;  206          0204  2     PTR = .NPARSE_BLOCK [NPA$L_FLDPTR];
;  207          0205  2
;  208          0206  2     NML$CHKPRMVAL (.NML$GL_PRMCODE, .LEN, .PTR);
```

```
;  209          0207  2     NML$SAVEPARAM (.NML$GL_PRMCODE, .LEN, .PTR);
;  210          0208  2
;  211          0209  2     NML$GL_PRMCODE = 0;                 ! Reset parsing code
;  212          0210  2
;  213          0211  2     RETURN NML$_STS_SUC
;  214          0212  2
;  215          0213  1     END;                                ! End of NML$PRM_CHECK


;                                                               .TITLE  NML$PARPRM NML NPARSE action routines for parsi
                                                                        ng paramete
                                                                .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$GW_PRMDESCNT
                                                                .EXTRN  NML$CHKEXE, NML$CHKPRMVAL
                                                                .EXTRN  NML$ERROR_2, NML$FIX_NODE_NUM
                                                                .EXTRN  NML$GETNODADR, NML$SAVEPARAM

                                                                .PSECT  $CODE$,NOWRT,2
```

```
                                        001C 00000          .ENTRY  NML$PRM_CHECK, Save R2,R3,R4                    ; 0151
                                54 00000000G   00   9E 00002 MOVAB   NML$GL_PRMCODE, R4                           ;
                                53         10  AC   D0 00009 MOVL    16(NPARSE_BLOCK), LEN                       ; 0203
                                52         14  AC   D0 0000D MOVL    20(NPARSE_BLOCK), PTR                       ; 0204
                                               52   DD 00011 PUSHL   PTR                                        ; 0206
                                               53   DD 00013 PUSHL   LEN                                        ;
                                               64   DD 00015 PUSHL   NML$GL_PRMCODE                            ;
                    00000000G   00             03   FB 00017 CALLS   #3, NML$CHKPRMVAL                         ;
                                               52   DD 0001E PUSHL   PTR                                        ; 0207
                                               53   DD 00020 PUSHL   LEN                                        ;
                                               64   DD 00022 PUSHL   NML$GL_PRMCODE                            ;
                    00000000G   00             03   FB 00024 CALLS   #3, NML$SAVEPARAM                         ;
                                               64   D4 0002B CLRL    NML$GL_PRMCODE                             ; 0209
                                50             01   D0 0002D MOVL    #1, R0                                     ; 0211
                                               04   00030 RET                                                    ; 0213

; Routine Size:  49 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  217          0214  1 %SBTTL 'NML$PRM_OBJPRV  Check object privilege parameter value'
;  218          0215  1 GLOBAL ROUTINE NML$PRM_OBJPRV =
;  219          0216  1
;  220          0217  1 !++
;  221          0218  1 ! FUNCTIONAL DESCRIPTION:
;  222          0219  1 !
;  223          0220  1 !       This is a NPARSE action routine that handles the object privilege
;  224          0221  1 !       mask.  The mask is generally a quadword but only the lower longword
;  225          0222  1 !       is currently used by VMS.  If this situation changes than this routine
;  226          0223  1 !       must also be changed.
;  227          0224  1 !
;  228          0225  1 ! FORMAL PARAMETERS:
;  229          0226  1 !
;  230          0227  1 !       NONE
;  231          0228  1 !
;  232          0229  1 ! IMPLICIT INPUTS.
;  233          0230  1 !
;  234          0231  1 !       NPARSE_BLOCK (pointed to by AP) contains the parsed parameter data.
;  235          0232  1 !           NPA$L_FLDCNT is the parameter length.
;  236          0233  1 !           NPA$L_FLDPTR is a pointer to the parameter in the received
;  237          0234  1 !               message buffer.
;  238          0235  1 !       NML$GL_PRMCODE contains the index into the change parameter table
;  239          0236  1 !           (NML$AB_CPTABLE).
;  240          0237  1 !
;  241          0238  1 ! IMPLICIT OUTPUTS:
;  242          0239  1 !
;  243          0240  1 !       If the parameter is valid then a descriptor entry will be created for
;  244          0241  1 !       it in NML$AW_PRM_DES and NML$GW_PRMDESCNT will be incremented.
;  245          0242  1 !
;  246          0243  1 ! ROUTINE VALUE:
;  247          0244  1 ! COMPLETION CODES:
;  248          0245  1 !
;  249          0246  1 !       Always returns NML$_STS_SUC.
;  250          0247  1 !
;  251          0248  1 ! SIDE EFFECTS:
;  252          0249  1 !
;  253          0250  1 !       If the parameter fails the semantic check, an error message
;  254          0251  1 !       (NMA$C_STS_PVA) is signalled.  If the parameter descriptor buffer
;  255          0252  1 !       is full then a software error (NMA$C_STS_MPR) is signalled with
;  256          0253  1 !       optional text to identify the error.
;  257          0254  1 !
;  258          0255  1 !--
;  259          0256  1
;  260          0257  2     BEGIN
;  261          0258  2
;  262          0259  2     $NPA_ARGDEF;                        ! Define NPARSE block reference
;  263          0260  2
;  264          0261  2     LOCAL
;  265          0262  2         MSGSIZE,                        ! Resultant message size
;  266          0263  2         LEN,
;  267          0264  2         PTR;                            ! Temporary parameter pointer
;  268          0265  2 !
;  269          0266  2 ! Add descriptor entry for this parameter.
;  270          0267  2 !
;  271          0268  2     LEN = .NPARSE_BLOCK [NPA$L_FLDCNT] - 1;
;  272          0269  2
;  273          0270  2     IF .LEN GTRU 4
```

```
;  274          0271  2     THEN
;  275          0272  2         LEN = 4;                          ! Maximum of four bytes
;  276          0273  2
;  277          0274  2     PTR = .NPARSE_BLOCK [NPA$L_FLDPTR] + 1;
;  278          0275  2     NML$SAVEPARAM (.NML$GL_PRMCODE, .LEN, .PTR);
;  279          0276  2
;  280          0277  2     NML$GL_PRMCODE = 0;                   ! Reset parsing code
;  281          0278  2
;  282          0279  2     RETURN NML$_STS_SUC
;  283          0280  2
;  284          0281  1     END;                                  ! End of NML$PRM_OBJPRV




                                                0004 00000         .ENTRY  NML$PRM_OBJPRV, Save R2                 ; 0215
                                52 00000000G    00   9E 00002     MOVAB   NML$GL_PRMCODE, R2                      ;
                      51        10    AC        01   C3 00009     SUBL3   #1, 16(NPARSE_BLOCK), LEN               ; 0268
                                04              51   D1 0000E     CMPL    LEN, #4                                 ; 0270
                                                03   1B 00011     BLEQU   1$                                      ;
                                51              04   D0 00013     MOVL    #4, LEN                                 ; 0272
                      50        14    AC        01   C1 00016 1$:  ADDL3   #1, 20(NPARSE_BLOCK), PTR             ; 0274
                                                50   DD 0001B     PUSHL   PTR                                     ; 0275
                                                51   DD 0001D     PUSHL   LEN                                     ;
                                                62   DD 0001F     PUSHL   NML$GL_PRMCODE                          ;
                         00000000G  00          03   FB 00021     CALLS   #3, NML$SAVEPARAM                       ;
                                                62   D4 00028     CLRL    NML$GL_PRMCODE                          ; 0277
                                50              01   D0 0002A     MOVL    #1, R0                                  ; 0279
                                                04   0002D       RET                                             ; 0281

; Routine Size:  46 bytes,    Routine Base:  $CODE$ + 0031
```

```
;  286          0282  1 %SBTTL 'NML$PRM_STRCHK  Check parameter value'
;  287          0283  1 GLOBAL ROUTINE NML$PRM_STRCHK =
;  288          0284  1
;  289          0285  1 !++
;  290          0286  1 ! FUNCTIONAL DESCRIPTION:
;  291          0287  1 !
;  292          0288  1 !       This is a NPARSE action routine that checks a parameter value
;  293          0289  1 !       against its description in the parameter semantic table.  If
;  294          0290  1 !       the parameter is valid, a descriptor entry is created.
;  295          0291  1 !
;  296          0292  1 ! FORMAL PARAMETERS:
;  297          0293  1 !
;  298          0294  1 !       NONE
;  299          0295  1 !                                                                       ;
;  300          0296  1 ! IMPLICIT INPUTS:
;  301          0297  1 !
;  302          0298  1 !       NPARSE_BLOCK (pointed to by AP) contains the parsed parameter data.
;  303          0299  1 !           NPA$L_FLDCNT is the parameter length.
;  304          0300  1 !           NPA$L_FLDPTR is a pointer to the parameter in the received
;  305          0301  1 !               message buffer.
;  306          0302  1 !       NML$GL_PRMCODE contains the index into the change parameter table
;  307          0303  1 !           (NML$AB_CPTABLE).
;  308          0304  1 !
;  309          0305  1 ! IMPLICIT OUTPUTS:
;  310          0306  1 !
;  311          0307  1 !       If the parameter is valid then a descriptor entry will be created for
;  312          0308  1 !       it in NML$AW_PRM_DES and NML$GW_PRMDESCNT will be incremented.
;  313          0309  1 !
;  314          0310  1 ! ROUTINE VALUE:
;  315          0311  1 ! COMPLETION CODES:
;  316          0312  1 !
;  317          0313  1 !       Always returns NML$_STS_SUC.
;  318          0314  1 !
;  319          0315  1 ! SIDE EFFECTS:
;  320          0316  1 !
;  321          0317  1 !       If the parameter fails the semantic check, an error message
;  322          0318  1 !       (NMA$C_STS_PVA) is signalled.  If the parameter descriptor buffer
;  323          0319  1 !       is full then a software error (NMA$C_STS_MPR) is signalled with
;  324          0320  1 !       optional text to identify the error.
;  325          0321  1 !
;  326          0322  1 !--
;  327          0323  1
;  328          0324  2     BEGIN
;  329          0325  2
;  330          0326  2     $NPA_ARGDEF;                        ! Define NPARSE block reference
;  331          0327  2
;  332          0328  2     LOCAL
;  333          0329  2         MSGSIZE,                        ! Resultant message size
;  334          0330  2         LEN,
;  335          0331  2         PTR;                            ! Temporary parameter pointer
;  336          0332  2 !
;  337          0333  2 ! Add descriptor entry for this parameter.
;  338          0334  2 !
;  339          0335  2     LEN = .NPARSE_BLOCK [NPA$L_FLDCNT] - 1;
;  340          0336  2     PTR = .NPARSE_BLOCK [NPA$L_FLDPTR] + 1;
;  341          0337  2
;  342          0338  2     NML$CHKPRMVAL (.NML$GL_PRMCODE, .LEN, .PTR);
```

```
;  343          0339  2     NML$SAVEPARAM (.NML$GL_PRMCODE, .LEN, .PTR);
;  344          0340  2
;  345          0341  2     NML$GL_PRMCODE = 0;                     ! Reset parsing code
;  346          0342  2
;  347          0343  2     RETURN NML$_STS_SUC
;  348          0344  2
;  349          0345  1     END;                                    ! End of NML$PRM_STRCHK


                                                 001C 00000        .ENTRY  NML$PRM_STRCHK, Save R2,R3,R4                ; 0283
                                       54 00000000G  00  9E 00002  MOVAB   NML$GL_PRMCODE, R4                           ;
                                 53        10    AC    01  C3 00009  SUBL3   #1, 16(NPARSE_BLOCK), LEN                  ; 0335
                                 52        14    AC    01  C1 0000E  ADDL3   #1, 20(NPARSE_BLOCK), PTR                  ; 0336
                                                       52  DD 00013  PUSHL   PTR                                      ; 0338
                                                       53  DD 00015  PUSHL   LEN                                      ;
                                                       64  DD 00017  PUSHL   NML$GL_PRMCODE                          ;
                                    00000000G  00      03  FB 00019  CALLS   #3, NML$CHKPRMVAL                        ;
                                                       52  DD 00020  PUSHL   PTR                                      ; 0339
                                                       53  DD 00022  PUSHL   LEN                                      ;
                                                       64  DD 00024  PUSHL   NML$GL_PRMCODE                          ;
                                    00000000G  00      03  FB 00026  CALLS   #3, NML$SAVEPARAM                        ;
                                                       64  D4 0002D  CLRL    NML$GL_PRMCODE                           ; 0341
                                               50      01  D0 0002F  MOVL    #1, R0                                   ; 0343
                                                       04  00032    RET                                              ; 0345

; Routine Size:  51 bytes,    Routine Base:  $CODE$ + 005F
```

```
;  351          0346  1 %SBTTL 'NML$PRM_CLEAR  Remove entity parameter from data base (action routine)'
;  352          0347  1 GLOBAL ROUTINE NML$PRM_CLEAR =
;  353          0348  1
;  354          0349  1 !++
;  355          0350  1 ! FUNCTIONAL DESCRIPTION:
;  356          0351  1 !
;  357          0352  1 !       This is a NPARSE action routine that removes a parameter from the
;  358          0353  1 !       specified entity by creating a parameter descriptor.
;  359          0354  1 !
;  360          0355  1 ! FORMAL PARAMETERS:
;  361          0356  1 !
;  362          0357  1 !       NONE
;  363          0358  1 !
;  364          0359  1 ! IMPLICIT INPUTS:
;  365          0360  1 !
;  366          0361  1 !       NPARSE argument block.
;  367          0362  1 !
;  368          0363  1 ! IMPLICIT OUTPUTS:
;  369          0364  1 !
;  370          0365  1 !       A parameter descriptor is added to NML$AW_PRM_DES and the count
;  371          0366  1 !       (NML$GW_PRMDESCNT) is incremented.
;  372          0367  1 !
;  373          0368  1 ! ROUTINE VALUE:
;  374          0369  1 ! COMPLETION CODES:
;  375          0370  1 !
;  376          0371  1 !       Always returns success (NML$_STS_SUC).
;  377          0372  1 !
;  378          0373  1 ! SIDE EFFECTS:
;  379          0374  1 !
;  380          0375  1 !       If no room remains in the descriptor area then a software error
;  381          0376  1 !       (NMA$C_STS_MPR) will be signalled indicating buffer overflow.
;  382          0377  1 !
;  383          0378  1 !--
;  384          0379  1
;  385          0380  2     BEGIN
;  386          0381  2
;  387          0382  2     $NPA_ARGDEF;                            ! Define NPARSE block reference
;  388          0383  2 !
;  389          0384  2 ! Add descriptor.
;  390          0385  2 !
;  391          0386  2     NML$GL_PRMCODE = .NPARSE_BLOCK [NPA$L_PARAM];
;  392          0387  2     NML$SAVEPARAM (.NML$GL_PRMCODE, 0, 0);
;  393          0388  2     NML$GL_PRMCODE = 0;
;  394          0389  2
;  395          0390  2     RETURN NML$_STS_SUC
;  396          0391  2
;  397          0392  1     END;                                    ! End of NML$PRM_CLEAR


                                0004 00000          .ENTRY  NML$PRM_CLEAR, Save R2                          ; 0347
                    52 00000000G  00    9E 00002    MOVAB   NML$GL_PRMCODE, R2                              ;
                    62        20  AC    D0 00009    MOVL    32(NPARSE_BLOCK), NML$GL_PRMCODE                ; 0386
                                  7E    7C 0000D    CLRQ    -(SP)                                           ; 0387
                                  62    DD 0000F    PUSHL   NML$GL_PRMCODE                                  ;
```

```
                         00000000G  00          03  FB 00011        CALLS   #3, NML$SAVEPARAM                        ;
                                    62          D4  00018        CLRL    NML$GL_PRMCODE                          ; 0388
                                    50          01  D0  0001A        MOVL    #1, R0                                  ; 0390
                                                    04  0001D        RET                                             ; 0392

; Routine Size:  30 bytes,    Routine Base:  $CODE$ + 0092

;  398          0393  1
```

```
;  400          0394  1 %SBTTL 'NML$PRM_CHKEXE  Check executor parameter group'
;  401          0395  1 GLOBAL ROUTINE NML$PRM_CHKEXE =
;  402          0396  1
;  403          0397  1 !++
;  404          0398  1 ! FUNCTIONAL DESCRIPTION:
;  405          0399  1 !
;  406          0400  1 !       This is a NPARSE action routine that checks the parameter
;  407          0401  1 !       code to make sure it is in the same group (executor-only or
;  408          0402  1 !       node) as any previously specified parameters.
;  409          0403  1 !
;  410          0404  1 ! FORMAL PARAMETERS:
;  411          0405  1 !
;  412          0406  1 !       NONE
;  413          0407  1 !
;  414          0408  1 ! IMPLICIT INPUTS:
;  415          0409  1 !
;  416          0410  1 !       NPARSE_BLOCK (pointed to by AP) contains the parsed parameter data.
;  417          0411  1 !           NPA$L_FLDPTR is a pointer code to the parameter in the received
;  418          0412  1 !               message buffer.
;  419          0413  1 !
;  420          0414  1 ! IMPLICIT OUTPUTS:
;  421          0415  1 !
;  422          0416  1 !       If the parameter is in the correct group then the flag for that
;  423          0417  1 !       group will be set in NML$GL_PRS_FLGS.
;  424          0418  1 !
;  425          0419  1 ! ROUTINE VALUE:
;  426          0420  1 ! COMPLETION CODES:
;  427          0421  1 !
;  428          0422  1 !       Always returns NML$_STS_SUC.
;  429          0423  1 !
;  430          0424  1 ! SIDE EFFECTS:
;  431          0425  1 !
;  432          0426  1 !       If the parameter is in the wrong group then an invalid parameter
;  433          0427  1 !       grouping applicable error (NMA$C_STS_PGP) will be signalled.
;  434          0428  1 !
;  435          0429  1 !--
;  436          0430  1
;  437          0431  2     BEGIN
;  438          0432  2
;  439          0433  2     $NPA_ARGDEF;                            ! Define NPARSE block reference
;  440          0434  2
;  441          0435  2     LOCAL
;  442          0436  2         MSGSIZE;
;  443          0437  2 !
;  444          0438  2 ! If this is not the executor-only parameter group return error.
;  445          0439  2 !
;  446          0440  2     IF .NML$GL_PRS_FLGS [NML$V_PRS_NODPG]
;  447          0441  2     THEN
;  448          0442  2         NML$ERROR_2 (NMA$C_STS_PGP,
;  449          0443  2                      .(.NPARSE_BLOCK [NPA$L_FLDPTR])<0,16>);
;  450          0444  2
;  451          0445  2     NML$GL_PRS_FLGS [NML$V_PRS_EXEPG] = 1; ! Set executor flag
;  452          0446  2
;  453          0447  2     RETURN NML$_STS_SUC
;  454          0448  2
;  455          0449  1     END;                                    ! End of NML$PRM_CHKEXE
```

```
                                                 0000 00000          .ENTRY  NML$PRM_CHKEXE, Save nothing                    ; 0395
                                          01  E1 00002          BBC     #1, NML$GL_PRS_FLGS+1, 1$                       ; 0440
                                    14    BC  3C 00C0A          MOVZWL  @20(NPARSE_BLOCK), -(SP)                        ; 0443
                                          1B  CE 0000E          MNEGL   #27, -(SP)                                      ; 0442
                                          02  FB 00011          CALLS   #2, NML$ERROR_2                                 ;
                                          01  88 00018 1$:      BISB2   #1, NML$GL_PRS_FLGS+1                           ; 0445
                                          01  D0 0001F          MOVL    #1, R0                                          ; 0447
                                              04 00022          RET                                                     ; 0449

; Routine Size:  35 bytes,    Routine Base:  $CODE$ + 00B0
```

```
;  457          0450  1 %SBTTL 'NML$PRM_CHKNOD  Check node parameter group'
;  458          0451  1 GLOBAL ROUTINE NML$PRM_CHKNOD =
;  459          0452  1
;  460          0453  1 !++
;  461          0454  1 ! FUNCTIONAL DESCRIPTION:
;  462          0455  1 !
;  463          0456  1 !       This is a NPARSE action routine that checks the executor parameter
;  464          0457  1 !       code to make sure it is in the same group (executor-only or
;  465          0458  1 !       node) as any previously specified parameters.
;  466          0459  1 !
;  467          0460  1 ! FORMAL PARAMETERS:
;  468          0461  1 !
;  469          0462  1 !       NONE
;  470          0463  1 !
;  471          0464  1 ! IMPLICIT INPUTS:
;  472          0465  1 !
;  473          0466  1 !       NPARSE_BLOCK (pointed to by AP) contains the parsed parameter data.
;  474          0467  1 !           NPA$L_FLDPTR is a pointer code to the parameter in the received
;  475          0468  1 !               message buffer.
;  476          0469  1 !
;  477          0470  1 ! IMPLICIT OUTPUTS:
;  478          0471  1 !
;  479          0472  1 !       If the parameter is in the correct group then the flag for that
;  480          0473  1 !       group will be set in NML$GL_PRS_FLGS.
;  481          0474  1 !
;  482          0475  1 ! ROUTINE VALUE:
;  483          0476  1 ! COMPLETION CODES:
;  484          0477  1 !
;  485          0478  1 !       Always returns NML$_STS_SUC.
;  486          0479  1 !
;  487          0480  1 ! SIDE EFFECTS:
;  488          0481  1 !
;  489          0482  1 !       If the parameter is in the wrong group then an invalid parameter
;  490          0483  1 !       grouping applicable error (NMA$C_STS_PGP) will be signalled.
;  491          0484  1 !
;  492          0485  1 !--
;  493          0486  1
;  494          0487  2     BEGIN
;  495          0488  2
;  496          0489  2     $NPA_ARGDEF;                            ! Define NPARSE block reference
;  497          0490  2
;  498          0491  2     LOCAL
;  499          0492  2         MSGSIZE;
;  500          0493  2 !
;  501          0494  2 ! If this is not the executor-only parameter group return error.
;  502          0495  2 !
;  503          0496  2     IF .NML$GL_PRS_FLGS [NML$V_PRS_EXEPG]
;  504          0497  2     THEN
;  505          0498  2         NML$ERROR_2 (NMA$C_STS_PGP,
;  506          0499  2                      .(.NPARSE_BLOCK [NPA$L_FLDPTR])<0,16>);
;  507          0500  2
;  508          0501  2     NML$GL_PRS_FLGS [NML$V_PRS_NODPG] = 1; ! Set node flag
;  509          0502  2
;  510          0503  2     RETURN NML$_STS_SUC
;  511          0504  2
;  512          0505  1     END;                                    ! End of NML$PRM_CHKNOD
```

```
                                                 0000 00000          .ENTRY  NML$PRM_CHKNOD, Save nothing                    ; 0451
                                        0E 00000000G   00   E9 00002 BLBC    NML$GL_PRS_FLGS+1, 1$                        ; 0496
                                        7E         14  BC   3C 00009 MOVZWL  @20(NPARSE_BLOCK), -(SP)                     ; 0499
                                        7E             1B   CE 0000D MNEGL   #27, -(SP)                                   ; 0498
                                00000000G   00         02   FB 00010 CALLS   #2, NML$ERROR_2                              ;
                                00000000G   00         02   88 00017 1$:    BISB2   #2, NML$GL_PRS_FLGS+1                ; 0501
                                        50             01   D0 0001E MOVL    #1, R0                                       ; 0503
                                                          04 00021 RET                                                     ; 0505

; Routine Size:  34 bytes,    Routine Base:  $CODE$ + 00D3
```

```
;  514          0506  1 %SBTTL 'NML$PRM_CHKREM  Check remote node parameter group'
;  515          0507  1 GLOBAL ROUTINE NML$PRM_CHKREM =
;  516          0508  1
;  517          0509  1 !++
;  518          0510  1 ! FUNCTIONAL DESCRIPTION:
;  519          0511  1 !
;  520          0512  1 !       This is a NPARSE action routine that checks the remote node parameter
;  521          0513  1 !       code to make sure it is in the same group (loop or remote
;  522          0514  1 !       node) as any previously specified parameters.
;  523          0515  1 !
;  524          0516  1 ! FORMAL PARAMETERS:
;  525          0517  1 !
;  526          0518  1 !       NONE
;  527          0519  1 !
;  528          0520  1 ! IMPLICIT INPUTS:
;  529          0521  1 !
;  530          0522  1 !       NPARSE_BLOCK (pointed to by AP) contains the parsed parameter data.
;  531          0523  1 !           NPA$L_FLDPTR is a pointer code to the parameter in the received
;  532          0524  1 !               message buffer.
;  533          0525  1 !
;  534          0526  1 ! IMPLICIT OUTPUTS:
;  535          0527  1 !
;  536          0528  1 !       If the parameter is in the correct group then the flag for that
;  537          0529  1 !       group will be set in NML$GL_PRS_FLGS.
;  538          0530  1 !
;  539          0531  1 ! ROUTINE VALUE:
;  540          0532  1 ! COMPLETION CODES:
;  541          0533  1 !
;  542          0534  1 !       Always returns NML$_STS_SUC.
;  543          0535  1 !
;  544          0536  1 ! SIDE EFFECTS:
;  545          0537  1 !
;  546          0538  1 !       If the parameter is in the wrong group then an invalid parameter
;  547          0539  1 !       grouping applicable error (NMA$C_STS_PGP) will be signalled.
;  548          0540  1 !
;  549          0541  1 !--
;  550          0542  1
;  551          0543  2     BEGIN
;  552          0544  2
;  553          0545  2     $NPA_ARGDEF;                        ! Define NPARSE block reference
;  554          0546  2
;  555          0547  2     LOCAL
;  556          0548  2         MSGSIZE;
;  557          0549  2 !
;  558          0550  2 ! If this is the loop node parameter group return error.
;  559          0551  2 !
;  560          0552  2     IF .NML$GL_PRS_FLGS [NML$V_PRS_LOOPG]
;  561          0553  2     THEN
;  562          0554  2         NML$ERROR_2 (NMA$C_STS_PNA,
;  563          0555  2                      .(.NPARSE_BLOCK [NPA$L_FLDPTR])<0,16>);
;  564          0556  2
;  565          0557  2     NML$GL_PRS_FLGS [NML$V_PRS_REMPG] = 1; ! Set remote node flag
;  566          0558  2
;  567          0559  2     RETURN NML$_STS_SUC
;  568          0560  2
;  569          0561  1     END;                                ! End of NML$PRM_CHKREM
```

```
                                                 0000 00000          .ENTRY  NML$PRM_CHKREM, Save nothing                    ; 0507
                    0E 00000000G   00                03  E1 00002    BBC     #3, NML$GL_PRS_FLGS+1, 1$                       ; 0552
                                   7E          14    BC  3C 0000A    MOVZWL  @20(NPARSE_BLOCK), -(SP)                        ; 0555
                                   7E                16  CE 0000E    MNEGL   #22, -(SP)                                      ; 0554
                       00000000G   00                02  FB 00011    CALLS   #2, NML$ERROR_2                                 ;
                       00000000G   00                04  88 00018 1$: BISB2   #4, NML$GL_PRS_FLGS+1                          ; 0557
                                   50                01  D0 0001F    MOVL    #1, R0                                          ; 0559
                                                     04  00022      RET                                                     ; 0561

; Routine Size:  35 bytes,    Routine Base:  $CODE$ + 00F5
```

```
;  571          0562  1 %SBTTL 'NML$PRM_CHKLOO  Check loop node parameter group'
;  572          0563  1 GLOBAL ROUTINE NML$PRM_CHKLOO =
;  573          0564  1
;  574          0565  1 !++
;  575          0566  1 ! FUNCTIONAL DESCRIPTION:
;  576          0567  1 !
;  577          0568  1 !       This is a NPARSE action routine that checks the loop node parameter
;  578          0569  1 !       code to make sure it is in the same group (remote or loop
;  579          0570  1 !       node) as any previously specified parameters.
;  580          0571  1 !
;  581          0572  1 ! FORMAL PARAMETERS:
;  582          0573  1 !
;  583          0574  1 !       NONE
;  584          0575  1 !
;  585          0576  1 ! IMPLICIT INPUTS:
;  586          0577  1 !
;  587          0578  1 !       NPARSE_BLOCK (pointed to by AP) contains the parsed parameter data.
;  588          0579  1 !           NPA$L_FLDPTR is a pointer code to the parameter in the received
;  589          0580  1 !               message buffer.
;  590          0581  1 !
;  591          0582  1 ! IMPLICIT OUTPUTS:
;  592          0583  1 !
;  593          0584  1 !       If the parameter is in the correct group then the flag for that
;  594          0585  1 !       group will be set in NML$GL_PRS_FLGS.
;  595          0586  1 !
;  596          0587  1 ! ROUTINE VALUE:
;  597          0588  1 ! COMPLETION CODES:
;  598          0589  1 !
;  599          0590  1 !       Always returns NML$_STS_SUC.
;  600          0591  1 !
;  601          0592  1 ! SIDE EFFECTS:
;  602          0593  1 !
;  603          0594  1 !       If the parameter is in the wrong group then a parameter not
;  604          0595  1 !       applicable error (NMA$C_STS_PNA) will be signalled.
;  605          0596  1 !
;  606          0597  1 !--
;  607          0598  1
;  608          0599  2     BEGIN
;  609          0600  2
;  610          0601  2     $NPA_ARGDEF;                        ! Define NPARSE block reference
;  611          0602  2
;  612          0603  2     MAP
;  613          0604  2         NML$GB_OPTIONS : BBLOCK [1];
;  614          0605  2
;  615          0606  2     EXTERNAL LITERAL
;  616          0607  2         CPT$GK_PCNO_A$$;                ! Loop node address
;  617          0608  2
;  618          0609  2     BIND
;  619          0610  2         NODADR = UPLIT WORD (0) : WORD;
;  620          0611  2 !
;  621          0612  2 ! If this is the remote node parameter group return error.
;  622          0613  2 !
;  623          0614  2     IF .NML$GL_PRS_FLGS [NML$V_PRS_REMPG]
;  624          0615  2     THEN
;  625          0616  2         NML$ERROR_2 (NMA$C_STS_PNA,
;  626          0617  2                      .(.NPARSE_BLOCK [NPA$L_FLDPTR])<0,16>);
;  627          0618  2
```

```
;  628          0619  2     NML$GL_PRS_FLGS [NML$V_PRS_LOOPG] = 1; ! Set loop node flag
;  629          0620  2 !
;  630          0621  2 ! If this is a SET/DEFINE (not CLEAR/PURGE) operation then add a zero
;  631          0622  2 ! node address to get around an idiosyncracy in NETACP.
;  632          0623  2 !
;  633          0624  2     IF NOT .NML$GB_OPTIONS [NMA$V_OPT_CLE]
;  634          0625  2     THEN
;  635          0626  2         NML$SAVEPARAM (CPT$GK_PCNO_A$$, 2, NODADR);
;  636          0627  2
;  637          0628  2     RETURN NML$_STS_SUC
;  638          0629  2
;  639          0630  1     END;                                ! End of NML$PRM_CHKLOO


                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

                                            0000  00000 P.AAA:  .WORD   0                                                ;

                                                  NODADR=       P.AAA
                                                                .EXTRN  CPT$GK_PCNO_A$$

                                                                .PSECT  $CODE$,NOWRT,2

                                              0000 00000        .ENTRY  NML$PRM_CHKLOO, Save nothing                      ; 0563
                  0E 00000000G  00          02  E1 00002        BBC     #2, NML$GL_PRS_FLGS+1, 1$                         ; 0614
                                7E      14  02  BC  3C 0000A        MOVZWL  @20(NPARSE_BLOCK), -(SP)                      ; 0617
                                7E          02  CE 0000E        MNEGL   #22, -(SP)                                        ; 0616
                     00000000G  00          02  FB 00011        CALLS   #2, NML$ERROR_2                                   ;
                     00000000G  00          02  88 00018 1$:     BISB2   #8, NML$GL_PRS_FLGS+1                             ; 0619
                  15 00000000G  00          02  E0 0001F        BBS     #6, NML$GB_OPTIONS, 2$                            ; 0624
                                    00000000' 00  9F 00027        PUSHAB  NODADR                                            ; 0626
                                              02  DD 0002D        PUSHL   #2                                                ;
                                    00000000G 8F  DD 0002F        PUSHL   #CPT$GK_PCNO_A$$                                  ;
                           00000000G  00          03  FB 00035        CALLS   #3, NML$SAVEPARAM                             ;
                                      50          01  D0 0003C 2$:     MOVL    #1, R0                                    ; 0628
                                                  04  0003F        RET                                                       ; 0630

; Routine Size:  64 bytes,    Routine Base:  $CODE$ + 0118
```

```
;  641          0631  1 %SBTTL 'NML$PRM_CHKKNO  Check for KNOWN entity option'
;  642          0632  1 GLOBAL ROUTINE NML$PRM_CHKKNO =
;  643          0633  1
;  644          0634  1 !++
;  645          0635  1 ! FUNCTIONAL DESCRIPTION:
;  646          0636  1 !
;  647          0637  1 !       This is a NPARSE action routine that checks the parameter
;  648          0638  1 !       code to make sure it is not applied to KNOWN parameters.
;  649          0639  1 !
;  650          0640  1 ! FORMAL PARAMETERS:
;  651          0641  1 !
;  652          0642  1 !       NONE
;  653          0643  1 !
;  654          0644  1 ! IMPLICIT INPUTS:
;  655          0645  1 !
;  656          0646  1 !       NPARSE_BLOCK (pointed to by AP) contains the parsed parameter data.
;  657          0647  1 !           NPA$L_FLDPTR is a pointer code to the parameter in the received
;  658          0648  1 !                        message buffer.
;  659          0649  1 !       NML$GB_ENTITY_FORMAT contains the entity format code.
;  660          0650  1 !
;  661          0651  1 ! IMPLICIT OUTPUTS:
;  662          0652  1 !
;  663          0653  1 !       NONE
;  664          0654  1 !
;  665          0655  1 ! ROUTINE VALUE:
;  666          0656  1 ! COMPLETION CODES:
;  667          0657  1 !
;  668          0658  1 !       Always returns NML$_STS_SUC.
;  669          0659  1 !
;  670          0660  1 ! SIDE EFFECTS:
;  671          0661  1 !
;  672          0662  1 !       If KNOWN entities is specified then a parameter not applicable
;  673          0663  1 !       error (NMA$C_STS_PNA) is signalled.
;  674          0664  1 !
;  675          0665  1 !--
;  676          0666  1
;  677          0667  2     BEGIN
;  678          0668  2
;  679          0669  2     $NPA_ARGDEF;                        ! Define NPARSE block reference
;  680          0670  2
;  681          0671  2     MAP
;  682          0672  2         NML$GB_ENTITY_FORMAT : BYTE SIGNED;
;  683          0673  2
;  684          0674  2     LOCAL
;  685          0675  2         MSGSIZE;
;  686          0676  2 !
;  687          0677  2 ! If KNOWN entities is selected then return error.
;  688          0678  2 !
;  689          0679  2     IF .NML$GB_ENTITY_FORMAT EQL NMA$C_ENT_KNO
;  690          0680  2     THEN
;  691          0681  2         NML$ERROR_2 (NMA$C_STS_PNA,
;  692          0682  2                      .(.NPARSE_BLOCK [NPA$L_FLDPTR])<0,16>);
;  693          0683  2
;  694          0684  2     RETURN NML$_STS_SUC
;  695          0685  2
;  696          0686  1     END;                                ! End of NML$PRM_CHKKNO
```

```
                                    0000 00000          .ENTRY  NML$PRM_CHKKNO, Save nothing          ; 0632
                FF   8F 00000000G  00   91 00002          CMPB    NML$GB_ENTITY_FORMAT, #-1            ; 0679
                                   0E   12 0000A          BNEQ    1$                                   ;
                     7E        14  BC   3C 0000C          MOVZWL  @20(NPARSE_BLOCK), -(SP)             ; 0682
                     7E            16   CE 00010          MNEGL   #22, -(SP)                           ; 0681
          00000000G  00            02   FB 00013          CALLS   #2, NML$ERROR_2                      ;
                     50            01   D0 0001A 1$:      MOVL    #1, R0                               ; 0684
                                        04 0001D          RET                                          ; 0686

; Routine Size:  30 bytes,    Routine Base:  $CODE$ + 0158
```

```
;  698          0687  1 %SBTTL 'NML$PRM_CHKEFI  Check event logging parameter group'
;  699          0688  1 GLOBAL ROUTINE NML$PRM_CHKEFI =
;  700          0689  1
;  701          0690  1 !++
;  702          0691  1 ! FUNCTIONAL DESCRIPTION:
;  703          0692  1 !
;  704          0693  1 !       This is a NPARSE action routine that checks the parameter
;  705          0694  1 !       code to make sure it is in the same group (event logging
;  706          0695  1 !       information) as any previously specified parameters.
;  707          0696  1 !
;  708          0697  1 ! FORMAL PARAMETERS:
;  709          0698  1 !
;  710          0699  1 !       NONE
;  711          0700  1 !
;  712          0701  1 ! IMPLICIT INPUTS:
;  713          0702  1 !
;  714          0703  1 !       NPARSE_BLOCK (pointed to by AP) contains the parsed parameter data.
;  715          0704  1 !           NPA$L_FLDPTR is a pointer code to the parameter in the received
;  716          0705  1 !               message buffer.
;  717          0706  1 !
;  718          0707  1 ! IMPLICIT OUTPUTS:
;  719          0708  1 !
;  720          0709  1 !       If the parameter is in the correct group then the flag for that
;  721          0710  1 !       group will be set in NML$GL_PRS_FLGS.
;  722          0711  1 !
;  723          0712  1 ! ROUTINE VALUE:
;  724          0713  1 ! COMPLETION CODES:
;  725          0714  1 !
;  726          0715  1 !       Always returns NML$_STS_SUC.
;  727          0716  1 !
;  728          0717  1 ! SIDE EFFECTS:
;  729          0718  1 !
;  730          0719  1 !       If the parameter is in the wrong group then an invalid parameter
;  731          0720  1 !       grouping applicable error (NMA$C_STS_PGP) will be signalled.
;  732          0721  1 !
;  733          0722  1 !--
;  734          0723  1
;  735          0724  2     BEGIN
;  736          0725  2
;  737          0726  2     $NPA_ARGDEF;                            ! Define NPARSE block reference
;  738          0727  2
;  739          0728  2     LOCAL
;  740          0729  2         MSGSIZE;
;  741          0730  2 !
;  742          0731  2 ! If this is not the event logging information parameter group return error.
;  743          0732  2 !
;  744          0733  2     IF .NML$GL_PRS_FLGS [NML$V_PRS_ESIPG]
;  745          0734  2     THEN
;  746          0735  2         NML$ERROR_2 (NMA$C_STS_PGP,
;  747          0736  2                      .(.NPARSE_BLOCK [NPA$L_FLDPTR])<0,16>);
;  748          0737  2
;  749          0738  2     NML$GL_PRS_FLGS [NML$V_PRS_EFIPG] = 1; ! Set group flag
;  750          0739  2
;  751          0740  2     RETURN NML$_STS_SUC
;  752          0741  2
;  753          0742  1     END;                                    ! End of NML$PRM_CHKEFI
```

```
                                        0000 00000          .ENTRY  NML$PRM_CHKEFI, Save nothing                    ; 0688
                    0E 000C0000G    00      04   E1 00002          BBC     #4, NML$GL_PRS_FLGS+1, 1$               ; 0733
                                    7E  14  BC   3C 0000A          MOVZWL  @20(NPARSE_BLOCK), -(SP)                ; 0736
                                    7E      1B   CE 0000E          MNEGL   #27, -(SP)                              ; 0735
                       00000000G    00      02   FB 00011          CALLS   #2, NML$ERROR_2                         ;
                       00000000G    00      08   88 00018 1$:      BISB2   #8, NML$GL_PRS_FLGS+1                   ; 0738
                                    50      01   D0 0001F          MOVL    #1, R0                                  ; 0740
                                            04 00022               RET                                             ; 0742

; Routine Size:  35 bytes,    Routine Base:  $CODE$ + 0176
```

```
;  755          0743  1 %SBTTL 'NML$PRM_CHKESI  Check event sink parameter group'
;  756          0744  1 GLOBAL ROUTINE NML$PRM_CHKESI =
;  757          0745  1
;  758          0746  1 !++
;  759          0747  1 ! FUNCTIONAL DESCRIPTION:
;  760          0748  1 !
;  761          0749  1 !       This is a NPARSE action routine that checks the parameter
;  762          0750  1 !       code to make sure it is in the same group (event sink
;  763          0751  1 !       information) as any previously specified parameters.
;  764          0752  1 !
;  765          0753  1 ! FORMAL PARAMETERS:
;  766          0754  1 !
;  767          0755  1 !       NONE
;  768          0756  1 !
;  769          0757  1 ! IMPLICIT INPUTS:
;  770          0758  1 !
;  771          0759  1 !       NPARSE_BLOCK (pointed to by AP) contains the parsed parameter data.
;  772          0760  1 !           NPA$L_FLDPTR is a pointer code to the parameter in the received
;  773          0761  1 !               message buffer.
;  774          0762  1 !
;  775          0763  1 ! IMPLICIT OUTPUTS:
;  776          0764  1 !
;  777          0765  1 !       If the parameter is in the correct group then the flag for that
;  778          0766  1 !       group will be set in NML$GL_PRS_FLGS.
;  779          0767  1 !
;  780          0768  1 ! ROUTINE VALUE:
;  781          0769  1 ! COMPLETION CODES:
;  782          0770  1 !
;  783          0771  1 !       Always returns NML$_STS_SUC.
;  784          0772  1 !
;  785          0773  1 ! SIDE EFFECTS:
;  786          0774  1 !
;  787          0775  1 !       If the parameter is in the wrong group then an invalid parameter
;  788          0776  1 !       grouping applicable error (NMA$C_STS_PGP) will be signalled.
;  789          0777  1 !
;  790          0778  1 !--
;  791          0779  1
;  792          0780  2     BEGIN
;  793          0781  2
;  794          0782  2     $NPA_ARGDEF;                            ! Define NPARSE block reference
;  795          0783  2
;  796          0784  2     LOCAL
;  797          0785  2         MSGSIZE;
;  798          0786  2 !
;  799          0787  2 ! If this is not the event sink information parameter group return error.
;  800          0788  2 !
;  801          0789  2     IF .NML$GL_PRS_FLGS [NML$V_PRS_EFIPG]
;  802          0790  2     THEN
;  803          0791  2         NML$ERROR_2 (NMA$C_STS_PGP,
;  804          0792  2                      .(.NPARSE_BLOCK [NPA$L_FLDPTR])<0,16>);
;  805          0793  2
;  806          0794  2     NML$GL_PRS_FLGS [NML$V_PRS_ESIPG] = 1; ! Set group flag
;  807          0795  2
;  808          0796  2     RETURN NML$_STS_SUC
;  809          0797  2
;  810          0798  1     END;                                    ! End of NML$PRM_CHKESI
```

```
                                    0000 00000          .ENTRY  NML$PRM_CHKESI, Save nothing                    ; 0744
                OE 00000000G    00          03   E1 00002       BBC     #3, NML$GL_PRS_FLGS+1, 1$               ; 0789
                                7E      14  BC   3C 0000A       MOVZWL  @20(NPARSE_BLOCK), -(SP)                ; 0792
                                7E          1B   CE 0000E       MNEGL   #27, -(SP)                              ; 0791
                00000000G       00          02   FB 00011       CALLS   #2, NML$ERROR_2                         ;
                00000000G       00          10   88 00018 1$:   BISB2   #16, NML$GL_PRS_FLGS+1                  ; 0794
                                50          01   D0 0001F       MOVL    #1, R0                                  ; 0796
                                            04 00022            RET                                             ; 0798

; Routine Size:  35 bytes,    Routine Base:  $CODE$ + 0199
```

```
;  812          0799  1 %SBTTL 'NML$PRM_CIRC_OWNER  Convert circuit owner parameter(action routine)'
;  813          0800  1 GLOBAL ROUTINE NML$PRM_CIRC_OWNER =
;  814          0801  1
;  815          0802  1 !++
;  816          0803  1 ! FUNCTIONAL DESCRIPTION:
;  817          0804  1 !       This is a NPARSE action routine that is called when parsing a
;  818          0805  1 !       SET CIRCUIT OWNER command.  It checks the node ID supplied
;  819          0806  1 !       to make sure it specifies the executor node (the only value
;  820          0807  1 !       currently allowed for OWNER).  This routine saves the OWNER
;  821          0808  1 !       parameter as a set bit field since only one value is allowed.
;  822          0809  1 !
;  823          0810  1 ! IMPLICIT INPUTS:
;  824          0811  1 !       NPARSE argument block.
;  825          0812  1 !               NPA$L_PARAM contains the error code.
;  826          0813  1 !               NPA$L_FLDPTR points to the parameter in the message.
;  827          0814  1 !
;  828          0815  1 ! ROUTINE VALUE:
;  829          0816  1 ! COMPLETION CODES:
;  830          0817  1 !       Signals NML$_STS_PVA if the node id is not the executor's
;  831          0818  1 !       (for now, we only support the executor as a circuit owner).
;  832          0819  1 !       Otherwise returns success.
;  833          0820  1 !
;  834          0821  1 !--
;  835          0822  1
;  836          0823  2 BEGIN
;  837          0824  2
;  838          0825  2 $NPA_ARGDEF;                    ! Define NPARSE block reference
;  839          0826  2
;  840          0827  2 LOCAL
;  841          0828  2     len,
;  842          0829  2     node_addr : WORD,
;  843          0830  2     name_addr;
;  844          0831  2
;  845          0832  2 node_addr = 0;
;  846          0833  2 len = .(.nparse_block [npa$l_fldptr])<0,8>;
;  847          0834  2 IF .nparse_block [npa$l_param] EQL nma$c_pcno_add THEN
;  848          0835  2     node_addr = .(.nparse_block [npa$l_fldptr]+1)<0,16>
;  849          0836  2 ELSE
;  850          0837  2     name_addr = .nparse_block [npa$l_fldptr] + 1;
;  851          0838  2 !
;  852          0839  2 ! If the node address in the NICE command matches the executor node
;  853          0840  2 ! address then set the flag to indicate it.
;  854          0841  2 !
;  855          0842  2 IF nml$chkexe (.nparse_block [npa$l_param], .node_addr, .len, .name_addr) THEN
;  856          0843  2     nml$saveparam (.nml$gl_prmcode, 1, UPLIT BYTE (1))
;  857          0844  2 ELSE
;  858          0845  2     nml$error_2 (nma$c_sts_pva,
;  859          0846  2                  nma$c_pcci_own);
;  860          0847  2 nml$gl_prmcode = 0;                          ! Reset parsing code
;  861          0848  2 RETURN nml$_sts_suc;                        ! Return status
;  862          0849  1 END;                          ! End of NML$PRM_CIRC_OWNER


                                                     .PSECT  $PLIT$,NOWRT,NOEXE,2

                                            01  00002 P.AAB:  .BYTE   1                                                ;
```

```
                                                                .PSECT  $CODE$,NOWRT,2

                                                 000C 00000     .ENTRY  NML$PRM_CIRC_OWNER, Save R2,R3                  ; 0800
                    53 00000000G     00   9E 00002     MOVAB   NML$GL_PRMCODE, R3                              ;
                                     52   B4 00009     CLRW    NODE_ADDR                                       ; 0832
                    50       14      AC   D0 0000B     MOVL    20(NPARSE_BLOCK), R0                            ; 0833
                    51               60   9A 0000F     MOVZBL  (R0), LEN                                       ;
000001F6            8F       20      AC   D1 00012     CMPL    32(NPARSE_BLOCK), #502                          ; 0834
                                     06   12 0001A     BNEQ    1$                                              ;
                    52       01      A0   B0 0001C     MOVW    1(R0), NODE_ADDR                                ; 0835
                                     02   11 00020     BRB     2$                                              ;
                                     50   D6 00022 1$:  INCL    NAME_ADDR                                       ; 0837
                                     50   DD 00024 2$:  PUSHL   NAME_ADDR                                       ; 0842
                                     51   DD 00026     PUSHL   LEN                                             ;
                    7E               52   3C 00028     MOVZWL  NODE_ADDR, -(SP)                                ;
                             20      AC   DD 0002B     PUSHL   32(NPARSE_BLOCK)                                ;
00000000G           00               04   FB 0002E     CALLS   #4, NML$CHKEXE                                  ;
                    13               50   E9 00035     BLBC    R0, 3$                                          ;
                       00000000'     00   9F 00038     PUSHAB  P.AAB                                           ; 0843
                                     01   DD 0003E     PUSHL   #1                                              ;
                                     63   DD 00040     PUSHL   NML$GL_PRMCODE                                  ;
00000000G           00               03   FB 00042     CALLS   #3, NML$SAVEPARAM                               ;
                                     0F   11 00049     BRB     4$                                              ;
                    7E       044C    8F   3C 0004B 3$:  MOVZWL  #1100, -(SP)                                   ; 0845
                    7E               10   CE 00050     MNEGL   #16, -(SP)                                      ;
00000000G           00               02   FB 00053     CALLS   #2, NML$ERROR_2                                 ;
                                     63   D4 0005A 4$:  CLRL    NML$GL_PRMCODE                                  ; 0847
                    50               01   D0 0005C     MOVL    #1, R0                                          ; 0848
                                          04 0005F     RET                                                     ; 0849

; Routine Size:  96 bytes,    Routine Base:  $CODE$ + 01BC
```

```
;  864     0850  1 %SBTTL 'NML$PRM_ERR  Build and signal error (action routine)'
;  865     0851  1 GLOBAL ROUTINE NML$PRM_ERR =
;  866     0852  1
;  867     0853  1 !++
;  868     0854  1 ! FUNCTIONAL DESCRIPTION:
;  869     0855  1 !
;  870     0856  1 !       This is a NPARSE action routine that signals parameter errors.
;  871     0857  1 !
;  872     0858  1 ! FORMAL PARAMETERS:
;  873     0859  1 !
;  874     0860  1 !       NONE
;  875     0861  1 !
;  876     0862  1 ! IMPLICIT INPUTS:
;  877     0863  1 !
;  878     0864  1 !       NPARSE argument block.
;  879     0865  1 !
;  880     0866  1 !               NPA$L_PARAM contains the error code.
;  881     0867  1 !               NPA$L_FLDPTR points to the parameter in the message.
;  882     0868  1 !
;  883     0869  1 ! IMPLICIT OUTPUTS:
;  884     0870  1 !
;  885     0871  1 !       NONE
;  886     0872  1 !
;  887     0873  1 ! ROUTINE VALUE:
;  888     0874  1 ! COMPLETION CODES:
;  889     0875  1 !
;  890     0876  1 !       Always returns success (NML$_STS_SUC).
;  891     0877  1 !
;  892     0878  1 ! SIDE EFFECTS:
;  893     0879  1 !
;  894     0880  1 !       Error message is signalled.
;  895     0881  1 !
;  896     0882  1 !--
;  897     0883  1
;  898     0884  2     BEGIN
;  899     0885  2
;  900     0886  2     $NPA_ARGDEF;                        ! Define NPARSE block reference
;  901     0887  2
;  902     0888  2     LOCAL
;  903     0889  2         ERR_CODE,                       ! Error code
;  904     0890  2         ERR_DETAIL;                     ! Error detail
;  905     0891  2
;  906     0892  2     ERR_CODE = .NPARSE_BLOCK [NPA$L_PARAM]; ! Get error code
;  907     0893  2 !
;  908     0894  2 ! Check for parameters to move in addition to error status.
;  909     0895  2 !
;  910     0896  3     ERR_DETAIL = (
;  911     0897  3         SELECTONEU .ERR_CODE OF
;  912     0898  3             SET
;  913     0899  3
;  914     0900  3             [NMA$C_STS_PTY,
;  915     0901  3              NMA$C_STS_PVA,
;  916     0902  3              NMA$C_STS_PNA]:
;  917     0903  3
;  918     0904  3                 .(.NPARSE_BLOCK [NPA$L_MSGPTR] - 2)<0,16,0>; ! Get detail code
;  919     0905  3
;  920     0906  3             [OTHERWISE]:
```

```
;  921          0907  3
;  922          0908  3                -1;
;  923          0909  3
;  924          0910  2             TES);
;  925          0911  2
;  926          0912  2     NML$ERROR_2 (.ERR_CODE, .ERR_DETAIL); ! Signal error message
;  927          0913  2
;  928          0914  2     RETURN NML$_STS_SUC                   ! Return success
;  929          0915  2
;  930          0916  1     END;                                  ! End of NML$PRM_ERR


                                        0000 00000          .ENTRY  NML$PRM_ERR, Save nothing                       ; 0851
                              51    20  AC  D0 00002          MOVL    32(NPARSE_BLOCK), ERR_CODE                      ; 0892
                FFFFFFEA      8F        51  D1 00005          CMPL    ERR_CODE, #-22                                  ; 0900
                                        12  13 00008          BEQL    1$                                              ;
                FFFFFFF0      8F        51  D1 0000A          CMPL    ERR_CODE, #-16                                  ;
                                        09  13 0000F          BEQL    1$                                              ;
                FFFFFFFA      8F        51  D1 00011          CMPL    ERR_CODE, #-6                                   ;
                                        0A  12 00018          BNEQ    2$                                              ;
                              50    08  AC  D0 0001A 1$:      MOVL    8(NPARSE_BLOCK), R0                             ; 0904
                              50    FE  A0  3C 0001E          MOVZWL  -2(R0), ERR_DETAIL                              ;
                                        03  11 00022          BRB     3$                                              ;
                              50        01  CE 00024 2$:      MNEGL   #1, ERR_DETAIL                                  ; 0908
                              50        50  DD 00027 3$:      PUSHL   ERR_DETAIL                                      ; 0912
                              51        51  DD 00029          PUSHL   ERR_CODE                                        ;
                00000000G     00        02  FB 0002B          CALLS   #2, NML$ERROR_2                                 ;
                              50        01  D0 00032          MOVL    #1, R0                                          ; 0914
                                        04  00035          RET                                                     ; 0916

; Routine Size:  61 bytes,    Routine Base:  $CODE$ + 021C

;  931          0917  1
```

```
;  933          0918  1 %SBTTL 'NML$PRM_EVTSRCTYP  Get event source type code'
;  934          0919  1 GLOBAL ROUTINE NML$PRM_EVTSRCTYP =
;  935          0920  1
;  936          0921  1 !++
;  937          0922  1 ! FUNCTIONAL DESCRIPTION:
;  938          0923  1 !
;  939          0924  1 !       This is an NPARSE action routine  that saves the event source type
;  940          0925  1 !       code.
;  941          0926  1 !
;  942          0927  1 ! FORMAL PARAMETERS:
;  943          0928  1 !
;  944          0929  1 !       NONE
;  945          0930  1 !
;  946          0931  1 ! IMPLICIT INPUTS:
;  947          0932  1 !
;  948          0933  1 !       NPARSE_BLOCK (pointed to by AP) contains the parsed parameter data.
;  949          0934  1 !           NPA$L_FLDCNT is the parameter length.
;  950          0935  1 !           NPA$L_FLDPTR is a pointer to the parameter in the received
;  951          0936  1 !               message buffer.
;  952          0937  1 !
;  953          0938  1 ! IMPLICIT OUTPUTS:
;  954          0939  1 !
;  955          0940  1 !
;  956          0941  1 ! ROUTINE VALUE:
;  957          0942  1 ! COMPLETION CODES:
;  958          0943  1 !
;  959          0944  1 !       Always returns success (NML$_STS_SUC).
;  960          0945  1 !
;  961          0946  1 ! SIDE EFFECTS:
;  962          0947  1 !
;  963          0948  1 !       NONE
;  964          0949  1 !
;  965          0950  1 !--
;  966          0951  1
;  967          0952  2     BEGIN
;  968          0953  2
;  969          0954  2     $NPA_ARGDEF;                        ! Define NPARSE block reference
;  970          0955  2 !
;  971          0956  2 ! Save the event source type code.
;  972          0957  2 !
;  973          0958  2     NML$GB_EVTSRCTYP = .(.NPARSE_BLOCK [NPA$L_FLDPTR])<0,8>;
;  974          0959  2     NML$GL_PRS_FLGS [NML$V_PRS_EVE] = 1; ! Flag event parameter processed
;  975          0960  2
;  976          0961  2     RETURN NML$_STS_SUC
;  977          0962  2
;  978          0963  1     END;                                ! End of NML$PRM_EVTSRCTYP


                                        0000 00000          .ENTRY  NML$PRM_EVTSRCTYP, Save nothing                 ; 0919
00000000G  00        14    BC  90 00002  MOVB    @20(NPARSE_BLOCK), NML$GB_EVTSRCTYP       ; 0958
00000000G  00             20  88 0000A  BISB2   #32, NML$GL_PRS_FLGS+1                    ; 0959
           50             01  D0 00011  MOVL    #1, R0                                    ; 0961
                              04 00014  RET                                               ; 0963
```

; Routine Size: 21 bytes,   Routine Base: $CODE$ + 0259

```
;  980          0964  1 %SBTTL 'NML$PRM_EVTSOURCE  Get event source descriptor'
;  981          0965  1 GLOBAL ROUTINE NML$PRM_EVTSOURCE =
;  982          0966  1
;  983          0967  1 !++
;  984          0968  1 ! FUNCTIONAL DESCRIPTION:
;  985          0969  1 !
;  986          0970  1 !       This is an NPARSE action routine that saves the event source id.
;  987          0971  1 !
;  988          0972  1 ! FORMAL PARAMETERS:
;  989          0973  1 !
;  990          0974  1 !       NONE
;  991          0975  1 !
;  992          0976  1 ! IMPLICIT INPUTS:
;  993          0977  1 !
;  994          0978  1 !       NPARSE_BLOCK (pointed to by AP) contains the parsed parameter data.
;  995          0979  1 !           NPA$L_FLDCNT is the parameter length.
;  996          0980  1 !           NPA$L_FLDPTR is a pointer to the parameter in the received
;  997          0981  1 !               message buffer.
;  998          0982  1 !
;  999          0983  1 ! IMPLICIT OUTPUTS:
; 1000          0984  1 !
; 1001          0985  1 !
; 1002          0986  1 ! ROUTINE VALUE:
; 1003          0987  1 ! COMPLETION CODES:
; 1004          0988  1 !
; 1005          0989  1 !       Always returns success (NML$_STS_SUC).
; 1006          0990  1 !
; 1007          0991  1 ! SIDE EFFECTS:
; 1008          0992  1 !
; 1009          0993  1 !       NONE
; 1010          0994  1 !
; 1011          0995  1 !--
; 1012          0996  1
; 1013          0997  2 BEGIN
; 1014          0998  2
; 1015          0999  2 $npa_argdef;                    ! Define NPARSE block reference
; 1016          1000  2
; 1017          1001  2 LOCAL
; 1018          1002  2     addr    : WORD,
; 1019          1003  2     namlen,
; 1020          1004  2     namptr;
; 1021          1005  2 !
; 1022          1006  2 ! Save the event source string descriptor information.
; 1023          1007  2 !
; 1024          1008  2 IF .nml$gb_evtsrctyp EQLU nma$c_ent_lin
; 1025          1009  2    OR .nml$gb_evtsrctyp EQLU nma$c_ent_cir THEN
; 1026          1010  3     BEGIN
; 1027          1011  3     nml$gq_evtsrcdsc [dsc$w_length] =
; 1028          1012  3                 .(.nparse_block [npa$l_fldptr])<0,8>;
; 1029          1013  3     nml$gq_evtsrcdsc [dsc$a_pointer] =
; 1030          1014  3                 .nparse_block [npa$l_fldptr] + 1;
; 1031          1015  3     END
; 1032          1016  2 ELSE
; 1033          1017  2     !
; 1034          1018  2     ! The event source is node.  Save the node id.  If the user specified
; 1035          1019  2     ! the node name (rather than the node address), get the node's address
; 1036          1020  2     ! from the DECnet node database.
```

```
;  1037          1021  2      !
;  1038          1022  3      BEGIN
;  1039          1023  3      IF .(.nparse_block [npa$l_fldptr])<0,8> EQLU 0 THEN
;  1040          1024  4          BEGIN
;  1041          1025  4          nml$gq_evtsrcdsc [dsc$a_pointer] =
;  1042          1026  4                  .(.nparse_block [npa$l_fldptr] + 1)<0,16>;
;  1043          1027  4          nml$fix_node_num (nml$gq_evtsrcdsc [dsc$a_pointer]);
;  1044          1028  4          END
;  1045          1029  3      ELSE
;  1046          1030  4          BEGIN
;  1047          1031  4          namlen = .(.nparse_block [npa$l_fldptr])<0,8>;
;  1048          1032  4          namptr = .nparse_block [npa$l_fldptr] + 1;
;  1049          1033  4
;  1050          1034  4          IF nml$getnodadr (.namlen, .namptr, addr) THEN
;  1051          1035  4              nml$gq_evtsrcdsc [dsc$a_pointer] = .addr
;  1052          1036  4          ELSE
;  1053          1037  4              nml$error_2 (nma$c_sts_ide, nma$c_ent_nod);
;  1054          1038  3          END;
;  1055          1039  3
;  1056          1040  3      nml$gq_evtsrcdsc [dsc$w_length] = 0;
;  1057          1041  2      END;
;  1058          1042  2
;  1059          1043  2  RETURN nml$_sts_suc
;  1060          1044  1  END;                                ! End of NML$PRM_EVTSOURCE


                                        0004 00000          .ENTRY  NML$PRM_EVTSOURCE, Save R2                          ; 0965
                              52 00000000G   00  9E 00002   MOVAB   NML$GQ_EVTSRCDSC+4, R2                               ;
                              5E             04  C2 00009   SUBL2   #4, SP                                             ;
                              50 00000000G   00  9A 0000C   MOVZBL  NML$GB_EVTSRCTYP, R0                                 ; 1008
                              01             50  91 00013   CMPB    R0, #1                                            ;
                                             05  13 00016   BEQL    1$                                                ;
                              03             50  91 00018   CMPB    R0, #3                                            ; 1009
                                             0C  12 0001B   BNEQ    2$                                                ;
                        FC    A2       14    BC  9B 0001D 1$:  MOVZBW  @20(NPARSE_BLOCK), NML$GQ_EVTSRCDSC               ; 1012
                  62    14    AC             01  C1 00022   ADDL3   #1, 20(NPARSE_BLOCK), NML$GQ_EVTSRCDSC+4           ; 1014
                                             3E  11 00027   BRB     6$                                                ; 1008
                              50       14    AC  D0 00029 2$:  MOVL    20(NPARSE_BLOCK), R0                              ; 1023
                                             60  95 0002D   TSTB    (R0)                                              ;
                                             0F  12 0002F   BNEQ    3$                                                ;
                              62       01    A0  3C 00031   MOVZWL  1(R0), NML$GQ_EVTSRCDSC+4                         ; 1026
                                             52  DD 00035   PUSHL   R2                                                ; 1027
                     00000000G   00          01  FB 00037   CALLS   #1, NML$FIX_NODE_NUM                               ;
                                             24  11 0003E   BRB     5$                                                ; 1023
                              51             80  9A 00040 3$:  MOVZBL  (R0)+, NAMLEN                                   ; 1031
                                   4001      8F  BB 00043   PUSHR   #^M<R0,SP>                                         ; 1034
                                             51  DD 00047   PUSHL   NAMLEN                                            ;
                     00000000G   00          03  FB 00049   CALLS   #3, NML$GETNODADR                                 ;
                              05             50  E9 00050   BLBC    R0, 4$                                            ;
                              62             6E  3C 00053   MOVZWL  ADDR, NML$GQ_EVTSRCDSC+4                          ; 1035
                                             0C  11 00056   BRB     5$                                                ;
                                             7E  D4 00058 4$:  CLRL    -(SP)                                             ; 1037
                              7E             09  CE 0005A   MNEGL   #9, -(SP)                                         ;
                     00000000G   00          02  FB 0005D   CALLS   #2, NML$ERROR_2                                   ;
```

```
                                    FC   A2  B4 00064 5$:     CLRW    NML$GQ_EVTSRCDSC                        ; 1040
                                50       01  D0 00067 6$:     MOVL    #1, R0                                  ; 1043
                                             04 0006A         RET                                             ; 1044

; Routine Size:  107 bytes,    Routine Base:  $CODE$ + 026E
```

```
;  1062          1045  1 %SBTTL 'NML$PRM_EVTCLASS  Get event class code'
;  1063          1046  1 GLOBAL ROUTINE NML$PRM_EVTCLASS =
;  1064          1047  1
;  1065          1048  1 !++
;  1066          1049  1 ! FUNCTIONAL DESCRIPTION:
;  1067          1050  1 !
;  1068          1051  1 !       This is a NPARSE action routine that saves the event class
;  1069          1052  1 !       information.
;  1070          1053  1 !
;  1071          1054  1 ! FORMAL PARAMETERS:
;  1072          1055  1 !
;  1073          1056  1 !       NONE
;  1074          1057  1 !
;  1075          1058  1 ! IMPLICIT INPUTS:
;  1076          1059  1 !
;  1077          1060  1 !       NPARSE_BLOCK (pointed to by AP) contains the parsed parameter data.
;  1078          1061  1 !           NPA$L_FLDCNT is the parameter length.
;  1079          1062  1 !           NPA$L_FLDPTR is a pointer to the parameter in the received
;  1080          1063  1 !               message buffer.
;  1081          1064  1 !
;  1082          1065  1 ! IMPLICIT OUTPUTS:
;  1083          1066  1 !
;  1084          1067  1 !
;  1085          1068  1 ! ROUTINE VALUE:
;  1086          1069  1 ! COMPLETION CODES:
;  1087          1070  1 !
;  1088          1071  1 !       Always returns success (NML$_STS_SUC).
;  1089          1072  1 !
;  1090          1073  1 ! SIDE EFFECTS:
;  1091          1074  1 !
;  1092          1075  1 !       NONE
;  1093          1076  1 !
;  1094          1077  1 !--
;  1095          1078  1
;  1096          1079  2     BEGIN
;  1097          1080  2
;  1098          1081  2     $NPA_ARGDEF;                            ! Define NPARSE block reference
;  1099          1082  2 !
;  1100          1083  2 ! Save the event class code (only the low 9 bits).
;  1101          1084  2 !
;  1102          1085  2     NML$GW_EVTCLASS = .(.NPARSE_BLOCK [NPA$L_FLDPTR])<0,9>;
;  1103          1086  2
;  1104          1087  2     RETURN NML$_STS_SUC
;  1105          1088  2
;  1106          1089  1     END;                                    ! End of NML$PRM_EVTCLASS



                                        0000 00000          .ENTRY  NML$PRM_EVTCLASS, Save nothing                  ; 1046
          50        14   BC                09          00   EF 00002          EXTZV   #0, #9, @20(NPARSE_BLOCK), R0   ; 1085
                    00000000G   00          50          50   B0 00008          MOVW    R0, NML$GW_EVTCLASS             ;
                                50          01          D0 0000F              MOVL    #1, R0                          ; 1087
                                                            04 00012          RET                                     ; 1089

; Routine Size:  19 bytes,    Routine Base:  $CODE$ + 02D9
```

```
;  1108          1090  1 %SBTTL 'NML$PRM_EVTMSKTYP  Get event mask type code'
;  1109          1091  1 GLOBAL ROUTINE NML$PRM_EVTMSKTYP =
;  1110          1092  1
;  1111          1093  1 !++
;  1112          1094  1 ! FUNCTIONAL DESCRIPTION:
;  1113          1095  1 !
;  1114          1096  1 !       This is a NPARSE action routine that saves the event mask type code.
;  1115          1097  1 !
;  1116          1098  1 ! FORMAL PARAMETERS:
;  1117          1099  1 !
;  1118          1100  1 !       NONE
;  1119          1101  1 !
;  1120          1102  1 ! IMPLICIT INPUTS:
;  1121          1103  1 !
;  1122          1104  1 !       NPARSE_BLOCK (pointed to by AP) contains the parsed parameter data.
;  1123          1105  1 !           NPA$L_FLDCNT is the parameter length.
;  1124          1106  1 !           NPA$L_FLDPTR is a pointer to the parameter in the received
;  1125          1107  1 !               message buffer.
;  1126          1108  1 !
;  1127          1109  1 ! IMPLICIT OUTPUTS:
;  1128          1110  1 !
;  1129          1111  1 !
;  1130          1112  1 ! ROUTINE VALUE:
;  1131          1113  1 ! COMPLETION CODES:
;  1132          1114  1 !
;  1133          1115  1 !       Always returns success (NML$_STS_SUC).
;  1134          1116  1 !
;  1135          1117  1 ! SIDE EFFECTS:
;  1136          1118  1 !
;  1137          1119  1 !       NONE
;  1138          1120  1 !
;  1139          1121  1 !--
;  1140          1122  1
;  1141          1123  2     BEGIN
;  1142          1124  2
;  1143          1125  2     $NPA_ARGDEF;                            ! Define NPARSE block reference
;  1144          1126  2 !
;  1145          1127  2 ! Save the event mask type code.
;  1146          1128  2 !
;  1147          1129  2     NML$GB_EVTMSKTYP = .(.NPARSE_BLOCK [NPA$L_FLDPTR])<6,2>;
;  1148          1130  2
;  1149          1131  2     RETURN NML$_STS_SUC
;  1150          1132  2
;  1151          1133  1     END;                                    ! End of NML$PRM_EVTMSKTYP


                                        0000 00000          .ENTRY  NML$PRM_EVTMSKTYP, Save nothing                 ; 1091
          50      14   BC                 02          06   EF 00002          EXTZV   #6, #2, @20(NPARSE_BLOCK), R0   ; 1129
                             00000000G    00          50   90 00008          MOVB    R0, NML$GB_EVTMSKTYP            ;
                                          50          01   D0 0000F          MOVL    #1, R0                          ; 1131
                                                           04 00012          RET                                     ; 1133

; Routine Size:  19 bytes,    Routine Base:  $CODE$ + 02EC
```

```
;  1153          1134  1 %SBTTL 'NML$PRM_EVTMASK  Get event mask'
;  1154          1135  1 GLOBAL ROUTINE NML$PRM_EVTMASK =
;  1155          1136  1
;  1156          1137  1 !++
;  1157          1138  1 ! FUNCTIONAL DESCRIPTION:
;  1158          1139  1 !
;  1159          1140  1 !       This is a NPARSE actior routine that saves the event mask.
;  1160          1141  1 !
;  1161          1142  1 ! FORMAL PARAMETERS:
;  1162          1143  1 !
;  1163          1144  1 !       NONE
;  1164          1145  1 !
;  1165          1146  1 ! IMPLICIT INPUTS:
;  1166          1147  1 !
;  1167          1148  1 !       NPARSE_BLOCK (pointed to by AP) contains the parsed parameter data.
;  1168          1149  1 !           NPA$L_FLDCNT is the parameter length.
;  1169          1150  1 !           NPA$L_FLDPTR is a pointer to the parameter in the received
;  1170          1151  1 !               message buffer.
;  1171          1152  1 !
;  1172          1153  1 ! IMPLICIT OUTPUTS:
;  1173          1154  1 !
;  1174          1155  1 !
;  1175          1156  1 ! ROUTINE VALUE:
;  1176          1157  1 ! COMPLETION CODES:
;  1177          1158  1 !
;  1178          1159  1 !       Always returns success (NML$_STS_SUC).
;  1179          1160  1 !
;  1180          1161  1 ! SIDE EFFECTS:
;  1181          1162  1 !
;  1182          1163  1 !       NONE
;  1183          1164  1 !
;  1184          1165  1 !--
;  1185          1166  1
;  1186          1167  2     BEGIN
;  1187          1168  2
;  1188          1169  2     $NPA_ARGDEF;                        ! Define NPARSE block reference
;  1189          1170  2
;  1190          1171  2     LOCAL
;  1191          1172  2         STATUS;                         ! Return status
;  1192          1173  2 !
;  1193          1174  2 ! Save the event mask descriptor information.
;  1194          1175  2 !
;  1195          1176  2     NML$GQ_EVTMSKDSC [DSC$W_LENGTH] =
;  1196          1177  2         .(.NPARSE_BLOCK [NPA$L_FLDPTR])<0,8>;
;  1197          1178  2
;  1198          1179  2     NML$GQ_EVTMSKDSC [DSC$A_POINTER] =
;  1199          1180  2         .NPARSE_BLOCK [NPA$L_FLDPTR] + 1;
;  1200          1181  2 !
;  1201          1182  2 ! If the event mask is all zeroes then it is not valid.
;  1202          1183  2 !
;  1203          1184  2     STATUS = NML$_STS_PVA;
;  1204          1185  2
;  1205          1186  2     INCR I FROM 0 TO .NML$GQ_EVTMSKDSC [DSC$W_LENGTH] - 1 DO
;  1206          1187  3         BEGIN
;  1207          1188  3
;  1208          1189  3         IF .(.NML$GQ_EVTMSKDSC [DSC$A_POINTER] + .I)<0,8> NEQ 0
;  1209          1190  3         THEN
```

```
;  1210          1191  4              BEGIN
;  1211          1192  4
;  1212          1193  4              STATUS = NML$_STS_SUC;
;  1213          1194  4              EXITLOOP;
;  1214          1195  4
;  1215          1196  3              END;
;  1216          1197  2          END;
;  1217          1198  2  !
;  1218          1199  2  ! If an error is detected then return an invalid parameter value error.
;  1219          1200  2  !
;  1220          1201  2      IF NOT .STATUS
;  1221          1202  2      THEN
;  1222          1203  2          NML$ERROR_2 (NMA$C_STS_PVA, NMA$C_PCLO_EVE);
;  1223          1204  2
;  1224          1205  2      RETURN .STATUS
;  1225          1206  2
;  1226          1207  1      END;                                ! End of NML$PRM_EVTMASK
```

```
                                                001C 00000          .ENTRY  NML$PRM_EVTMASK, Save R2,R3,R4                  ; 1135
                                    54 00000000G   00   9E 00002    MOVAB   NML$GQ_EVTMSKDSC, R4                            ;
                                    64         14  BC   9B 00009    MOVZBW  @20(NPARSE_BLOCK), NML$GQ_EVTMSKDSC              ; 1177
                                04  A4         14  AC   01   C1 0000D  ADDL3   #1, 20(NPARSE_BLOCK), NML$GQ_EVTMSKDSC+4   ; 1180
                                    53             20   CE 00013    MNEGL   #32, STATUS                                     ; 1184
                                    52             64   3C 00016    MOVZWL  NML$GQ_EVTMSKDSC, R2                            ; 1186
                                    51             01   CE 00019    MNEGL   #1, I                                           ; 1189
                                                   0E   11 0001C    BRB     2$                                              ;
                                    50         04  A4   D0 0001E 1$: MOVL    NML$GQ_EVTMSKDSC+4, R0                      ;
                                                6140   95 00022    TSTB    (I)[R0]                                         ;
                                                   05   13 00025    BEQL    2$                                              ;
                                    53             01   D0 00027    MOVL    #1, STATUS                                      ; 1193
                                                   04   11 0002A    BRB     3$                                              ; 1191
                                EE             51   52   F2 0002C 2$: AOBLSS  R2, I, 1$                                  ; 1186
                                               0E   53   E8 00030 3$: BLBS    STATUS, 4$                                      ; 1201
                                               7E   C9   8F   9A 00033    MOVZBL  #201, -(SP)                             ; 1203
                                               7E        10   CE 00037    MNEGL   #16, -(SP)                              ;
                                00000000G      C0        02   FB 0003A    CALLS   #2, NML$ERROR_2                         ;
                                               50        53   D0 00041 4$: MOVL    STATUS, R0                              ; 1205
                                                          04   00044    RET                                             ; 1207

; Routine Size.  69 bytes,    Routine Base:  $CODE$ + 02FF
```

```
;  1228          1208  1 %SBTTL 'NML$PRM_CHKEVE  Check for event parameter'
;  1229          1209  1 GLOBAL ROUTINE NML$PRM_CHKEVE =
;  1230          1210  1
;  1231          1211  1 !++
;  1232          1212  1 ! FUNCTIONAL DESCRIPTION:
;  1233          1213  1 !
;  1234          1214  1 !       This is a NPARSE action routine that checks for the presence
;  1235          1215  1 !       of the event parameter.
;  1236          1216  1 !
;  1237          1217  1 ! FORMAL PARAMETERS:
;  1238          1218  1 !
;  1239          1219  1 !       NONE
;  1240          1220  1 !
;  1241          1221  1 ! IMPLICIT INPUTS:
;  1242          1222  1 !
;  1243          1223  1 !       NPARSE_BLOCK
;  1244          1224  1 !
;  1245          1225  1 ! IMPLICIT OUTPUTS:
;  1246          1226  1 !
;  1247          1227  1 !       NONE
;  1248          1228  1 !
;  1249          1229  1 ! ROUTINE VALUE:
;  1250          1230  1 ! COMPLETION CODES:
;  1251          1231  1 !
;  1252          1232  1 !       Always returns NML$_STS_SUC.
;  1253          1233  1 !
;  1254          1234  1 ! SIDE EFFECTS:
;  1255          1235  1 !
;  1256          1236  1 !
;  1257          1237  1 !--
;  1258          1238  1
;  1259          1239  2     BEGIN
;  1260          1240  2
;  1261          1241  2     $NPA_ARGDEF;                            ! Define NPARSE block reference
;  1262          1242  2 !
;  1263          1243  2 ! If this is the event filter (EFI) parameter group and no event logging
;  1264          1244  2 ! parameter has been specified then return a parameter missing error.
;  1265          1245  2 !
;  1266          1246  2     IF .NML$GL_PRS_FLGS [NML$V_PRS_EFIPG]
;  1267          1247  2     AND NOT .NML$GL_PRS_FLGS [NML$V_PRS_EVE]
;  1268          1248  2     THEN
;  1269          1249  2         NML$ERROR_2 (NMA$C_STS_PMS,
;  1270          1250  2                      NMA$C_PCLO_EVE);
;  1271          1251  2
;  1272          1252  2     RETURN NML$_STS_SUC
;  1273          1253  2
;  1274          1254  1     END;                                    ! End of NML$PRM_CHKEVE


                                        0000 00000          .ENTRY  NML$PRM_CHKEVE, Save nothing                    ; 1209
                16 00000000G   00         03   E1 00002    BBC     #3, NML$GL_PRS_FLGS+1, 1$                       ; 1246
                0E 00000000G   00         05   E0 0000A    BBS     #5, NML$GL_PRS_FLGS+1, 1$                       ; 1247
                               7E    C9   8F   9A 00012    MOVZBL  #201, -(SP)                                     ; 1249
                               7E         1D   CE 00016    MNEGL   #29, -(SP)                                      ;
```

```
                 00000000G  00          02  FB 00019          CALLS   #2, NML$ERROR_2                        ;
                            50          01  D0 00020 1$:      MOVL    #1, R0                                ; 1252
                                            04 00023          RET                                           ; 1254

; Routine Size:  36 bytes,    Routine Base:  $CODE$ + 0344
```

```
;  1276          1255  1 %SBTTL 'NML$PRM_SAVE_NODE       Check and save node parameter'
;  1277          1256  1 GLOBAL ROUTINE NML$PRM_SAVE_NODE =
;  1278          1257  1
;  1279          1258  1 !++
;  1280          1259  1 ! FUNCTIONAL DESCRIPTION:
;  1281          1260  1 !
;  1282          1261  1 !       This is a NPARSE action that checks a node id for validity.
;  1283          1262  1 !       Some node parameters are saved in the DECnet databases as a node
;  1284          1263  1 !       address.  If the node is specified by name and no corresponding
;  1285          1264  1 !       address can be found in the data base then an error is returned.
;  1286          1265  1 !
;  1287          1266  1 ! FORMAL PARAMETERS:
;  1288          1267  1 !
;  1289          1268  1 !       NONE
;  1290          1269  1 !
;  1291          1270  1 ! IMPLICIT INPUTS:
;  1292          1271  1 !
;  1293          1272  1 !       NPARSE_BLOCK [NPA$L_FLDPTR] contains the pointer to the entity
;  1294          1273  1 !           format code and id string.
;  1295          1274  1 !
;  1296          1275  1 ! IMPLICIT OUTPUTS:
;  1297          1276  1 !
;  1298          1277  1 !       NONE
;  1299          1278  1 !
;  1300          1279  1 ! ROUTINE VALUE:
;  1301          1280  1 ! COMPLETION CODES:
;  1302          1281  1 !
;  1303          1282  1 !       Always returns success (NML$_STS_SUC).
;  1304          1283  1 !
;  1305          1284  1 ! SIDE EFFECTS:
;  1306          1285  1 !
;  1307          1286  1 !       If the host node id is invalid then an invalid identification format
;  1308          1287  1 !       error (NML$_STS_IDE) will be signalled.
;  1309          1288  1 !
;  1310          1289  1 !--
;  1311          1290  1
;  1312          1291  2 BEGIN
;  1313          1292  2
;  1314          1293  2 $npa_argdef;                        ! Define NPARSE block reference
;  1315          1294  2
;  1316          1295  2 OWN
;  1317          1296  2     node_addr : WORD;
;  1318          1297  2
;  1319          1298  2 LOCAL
;  1320          1299  2     length,
;  1321          1300  2     addr;
;  1322          1301  2
;  1323          1302  2 !
;  1324          1303  2 ! The NODE ADDRESS parameter is never specified as a name, and therefore,
;  1325          1304  2 ! its NICE parameter type is a word.
;  1326          1305  2 !
;  1327          1306  2 IF .nparse_block [npa$l_param] EQL nml$c_node_num_param THEN
;  1328          1307  3     BEGIN
;  1329          1308  3     length = 0;
;  1330          1309  3     addr = .nparse_block [npa$l_fldptr];
;  1331          1310  3     END
;  1332          1311  2 ELSE
```

```
;  1333          1312  2 !
;  1334          1313  2 ! Other node parameters are saved in address form, but can be specified
;  1335          1314  2 ! by either the name or the address.  So the NICE format is node-id.
;  1336          1315  2 ! If leading byte of the parameter (the length) is a 0, the user typed a
;  1337          1316  2 ! node address rather than counted node name string.  Get the length and
;  1338          1317  2 ! address of the node parameter.
;  1339          1318  2 !
;  1340          1319  3     BEGIN
;  1341          1320  3     length = .(.nparse_block [npa$l_fldptr])<0,8>; ! Get length
;  1342          1321  3     addr = .nparse_block [npa$l_fldptr] + 1;
;  1343          1322  2     END;
;  1344          1323  2 !
;  1345          1324  2 ! If length is zero then id is a node address, otherwise it is a counted
;  1346          1325  2 ! node name string.
;  1347          1326  2 !
;  1348          1327  2 IF .length EQL 0 THEN
;  1349          1328  3     BEGIN
;  1350          1329  3     node_addr = .(.addr)<0,16>;
;  1351          1330  3         !
;  1352          1331  3         ! If the node area was specified as zero, change to to 1 (if NCP
;  1353          1332  3         ! is a Phase IV NCP) or the executor's area (if NCP is a Phase III
;  1354          1333  3         ! NCP).
;  1355          1334  3         !
;  1356          1335  3     nml$fix_node_num (node_addr);
;  1357          1336  3     END
;  1358          1337  2 ELSE
;  1359          1338  2         !
;  1360          1339  2         ! The parameter was specified as a name.  Look the node up in
;  1361          1340  2         ! the node database to get it's address.
;  1362          1341  2         !
;  1363          1342  3     BEGIN
;  1364          1343  3     IF NOT nml$getnodadr (.length, .addr, node_addr) THEN
;  1365          1344  3         nml$error_2 (nma$c_sts_ide, nma$c_ent_nod);
;  1366          1345  2     END;
;  1367          1346  2 !
;  1368          1347  2 ! Save the node address parameter.
;  1369          1348  2 !
;  1370          1349  2 nml$saveparam (.nml$gl_prmcode, 2, node_addr);
;  1371          1350  2
;  1372          1351  2 nml$gl_prmcode = 0;                    ! Reset parameter code
;  1373          1352  2 RETURN nml$_sts_suc
;  1374          1353  1 END;                                ! End of NML$PRM_SAVE_NODE


                                                .PSECT  $OWN$,NOEXE,2

                                    00000 NODE_ADDR:
                                                .BLKB   2


                                                .PSECT  $CODE$,NOWRT,2

                                      000C 00000        .ENTRY  NML$PRM_SAVE_NODE, Save R2,R3                   ; 1256
                    53 00000000G   00   9E 00002        MOVAB   NML$GL_PRMCODE, R3                              ;
                    52 00000000'   00   9E 00009        MOVAB   NODE_ADDR, R2                                   ;
```

```
                                          20   AC  D5 00010          TSTL    32(NPARSE_BLOCK)                        ; 1306
                                               08  12 00013          BNEQ    1$
                                               51  D4 00015          CLRL    LENGTH                                  ; 1308
                                     50   14   AC  D0 00017          MOVL    20(NPARSE_BLOCK), ADDR                  ; 1309
                                               09  11 0001B          BRB     2$                                      ; 1306
                                     51   14   BC  9A 0001D 1$:      MOVZBL  @20(NPARSE_BLOCK), LENGTH               ; 1320
                          50    14   AC        01  C1 00021          ADDL3   #1, 20(NPARSE_BLOCK), ADDR              ; 1321
                                               51  D5 00026 2$:      TSTL    LENGTH                                  ; 1327
                                               0E  12 00028          BNEQ    3$
                                     62        60  B0 0002A          MOVW    (ADDR), NODE_ADDR                       ; 1329
                                               52  DD 0002D          PUSHL   R2                                      ; 1335
                               00000000G 00    01  FB 0002F          CALLS   #1, NML$FIX_NODE_NUM
                                               1A  11 00036          BRB     4$                                      ; 1327
                                               05  BB 00038 3$:      PUSHR   #^M<R0,R2>                              ; 1343
                                               51  DD 0003A          PUSHL   LENGTH
                               00000000G 00    03  FB 0003C          CALLS   #3, NML$GETNODADR
                                         0C    50  E8 00043          BLBS    R0, 4$
                                               7E  D4 00046          CLRL    -(SP)                                   ; 1344
                                         7E    09  CE 00048          MNEGL   #9, -(SP)
                               00000000G 00    02  FB 0004B          CALLS   #2, NML$ERROR_2
                                               52  DD 00052 4$:      PUSHL   R2                                      ; 1349
                                               02  DD 00054          PUSHL   #2
                                               63  DD 00056          PUSHL   NML$GL_PRMCODE
                               00000000G 00    03  FB 00058          CALLS   #3, NML$SAVEPARAM
                                               63  D4 0005F          CLRL    NML$GL_PRMCODE                          ; 1351
                                     50        01  D0 00061          MOVL    #1, R0                                  ; 1352
                                               04  00064          RET                                             ; 1353

; Routine Size:  101 bytes,    Routine Base:  $CODE$ + 0368
```

```
;  1376          1354  1 %SBTTL 'NML$PRM_SET_NET  Set X25-Protocol network ID'
;  1377          1355  1 GLOBAL ROUTINE NML$PRM_SET_NET =
;  1378          1356  1
;  1379          1357  1 !++
;  1380          1358  1 ! FUNCTIONAL DESCRIPTION:
;  1381          1359  1 !       This is a NPARSE action routine that is called only when NML
;  1382          1360  1 !       gets a NICE command of SET X25-PROTOCOL NET <net-id>.  In this
;  1383          1361  1 !       case only, the entity ID passed to the ACP must be the network
;  1384          1362  1 !       name.  In all other cases, the network ID is a zero length string
;  1385          1363  1 !       to indicate the currently active network.  SET X25-PROTOCOL NET
;  1386          1364  1 !       is used to specify the active network in the first place.
;  1387          1365  1 !
;  1388          1366  1 ! IMPLICIT INPUTS:
;  1389          1367  1 !
;  1390          1368  1 !       NPARSE_BLOCK (pointed to by AP) contains the parsed parameter data.
;  1391          1369  1 !           NPA$L_FLDCNT is the parameter length.
;  1392          1370  1 !           NPA$L_FLDPTR is a pointer to the parameter in the received
;  1393          1371  1 !               message buffer.
;  1394          1372  1 !       NML$GL_PRMCODE contains the index into the change parameter table
;  1395          1373  1 !           (NML$AB_CPTABLE).
;  1396          1374  1 !
;  1397          1375  1 ! ROUTINE VALUE:
;  1398          1376  1 ! COMPLETION CODES:
;  1399          1377  1 !       Always returns NML$_STS_SUC.
;  1400          1378  1 !
;  1401          1379  1 !--
;  1402          1380  1
;  1403          1381  2 BEGIN
;  1404          1382  2
;  1405          1383  2 MAP
;  1406          1384  2     NML$GB_OPTIONS : BBLOCK [1];
;  1407          1385  2
;  1408          1386  2 !
;  1409          1387  2 ! The entity ID has already been set up to be the active network
;  1410          1388  2 ! (NML$GB_ENTITY_FORMAT = 0) during initial parsing.  So, if the NICE
;  1411          1389  2 ! command is DEFINE, leave things as they are.  If it's a SET, pop
;  1412          1390  2 ! the network name off the parameter stack and save it as the entity ID
;  1413          1391  2 ! for the QIO P2 buffer.
;  1414          1392  2 !
;  1415          1393  2 IF NOT .NML$GB_OPTIONS [NMA$V_OPT_PER] THEN
;  1416          1394  3     BEGIN
;  1417          1395  3     !
;  1418          1396  3     ! Back up to the network parameter.  This pops the network parameter
;  1419          1397  3     ! off the parameter stack.
;  1420          1398  3     !
;  1421          1399  3     NML$GW_PRMDESCNT = .NML$GW_PRMDESCNT - 1;
;  1422          1400  3     !
;  1423          1401  3     ! Move the network ID to the entity ID globals.
;  1424          1402  3     !
;  1425          1403  3     NML$GB_ENTITY_FORMAT = .NML$AW_PRM_DES [.NML$GW_PRMDESCNT, PDB$W_COUNT];
;  1426          1404  3     CH$MOVE (.NML$GB_ENTITY_FORMAT,
;  1427          1405  3                 .NML$AW_PRM_DES [.NML$GW_PRMDESCNT, PDB$A_POINTER],
;  1428          1406  3                 NML$AB_ENTITY_ID);
;  1429          1407  2     END;
;  1430          1408  2
;  1431          1409  2 RETURN NML$_STS_SUC
;  1432          1410  2
```

```
; 1433          1411  1 END;                          ! End of NML$PRM_SET_NET



                                                00FC 00000          .ENTRY  NML$PRM_SET_NET, Save R2,R3,R4,R5,R6,R7     ; 1355
                                    57 00000000G  00  9E 00002      MOVAB   NML$GB_ENTITY_FORMAT, R7                    ;
                                    56 00000000G  00  9E 00009      MOVAB   NML$GW_PRMDESCNT, R6                        ;
                                       00000000G  00  95 00010      TSTB    NML$GB_OPTIONS                              ; 1393
                                                  24  19 00016      BLSS    1$                                          ;
                                                  66  B7 00018      DECW    NML$GW_PRMDESCNT                            ; 1399
                                    50            66  3C 0001A      MOVZWL  NML$GW_PRMDESCNT, R0                        ; 1403
                                       00000000G0040  7F 0001D      PUSHAQ  NML$AW_PRM_DES+2[R0]                        ;
                                    67            9E  90 00024      MOVB    @(SP)+, NML$GB_ENTITY_FORMAT                ;
                                    51            67  9A 00027      MOVZBL  NML$GB_ENTITY_FORMAT, R1                    ; 1404
                                       00000000G0040  7F 0002A      PUSHAQ  NML$AW_PRM_DES+4[R0]                        ; 1405
                                    50            9E  D0 00031      MOVL    @(SP)+, R0                                  ;
            00000000G  00           60            51  28 00034      MOVC3   R1, (R0), NML$AB_ENTITY_ID                  ; 1404
                                    50            01  D0 0003C 1$:   MOVL    #1, R0                                      ; 1409
                                                  04  0003F          RET                                             ; 1411

; Routine Size:  64 bytes,    Routine Base:  $CODE$ + 03CD
```

```
;1435    1412  1 %SBTTL 'NML$PRM_QUAL_FORMAT  Check X25-Protocol Group DTE format'
;1436    1413  1 GLOBAL ROUTINE NML$PRM_QUAL_FORMAT =
;1437    1414  1
;1438    1415  1 !++
;1439    1416  1 ! FUNCTIONAL DESCRIPTION:
;1440    1417  1 !       This is a NPARSE action routine that is called only when NML
;1441    1418  1 !       gets a NICE SET/DEFINE X25 PROTOCOL GROUP commard.
;1442    1419  1 !       In this case some SET/DEFINE commands are illegal, because
;1443    1420  1 !       with groups, you must always set DTE, number, and type
;1444    1421  1 !       together.  Also, the same DTE will have different group numbers in
;1445    1422  1 !       different groups but DTEs in the same group must have the same group
;1446    1423  1 !       number.
;1447    1424  1 !
;1448    1425  1 ! IMPLICIT INPUTS:
;1449    1426  1 !
;1450    1427  1 !       NPARSE_BLOCK (pointed to by AP) contains the parsed parameter data.
;1451    1428  1 !           NPA$L_FLDCNT is the parameter length.
;1452    1429  1 !           NPA$L_FLDPTR is a pointer to the parameter in the received
;1453    1430  1 !               message buffer.
;1454    1431  1 !
;1455    1432  1 ! ROUTINE VALUE:
;1456    1433  1 ! COMPLETION CODES:
;1457    1434  1 !
;1458    1435  1 !--
;1459    1436  1
;1460    1437  2 BEGIN
;1461    1438  2
;1462    1439  2 MAP
;1463    1440  2     NML$GB_OPTIONS : BBLOCK [1];
;1464    1441  2 !
;1465    1442  2 ! If KNOWN GROUPS allow only
;1466    1443  2 !       SET KNOWN GROUPS KNOWN DTES ALL or
;1467    1444  2 !       SET KNOWN GROUPS ALL
;1468    1445  2 !
;1469    1446  2 IF .NML$GB_ENTITY_FORMAT LEQ 0 THEN              ! If KNOWN grouos and
;1470    1447  3     BEGIN
;1471    1448  4     IF (.NML$GB_OPTIONS [NMA$V_OPT_PER] OR       ! not SET ALL
;1472    1449  3        NOT .NML$GL_PRS_FLGS [NML$V_PRS_ALL]) OR !       or
;1473    1450  4        (.NML$GL_PRS_FLGS [NML$V_PRS_QUALIFIER] AND ! DTE nnnn
;1474    1451  3         .NML$GB_QUALIFIER_FORMAT GTR 0) THEN
;1475    1452  3         RETURN NML$_STS_IDE;                     !       Error.
;1476    1453  3     END
;1477    1454  2 ELSE
;1478    1455  2     !
;1479    1456  2     ! If a specific group, allow only
;1480    1457  2     !   SET/DEFINE GROUP FRED DTE 333444555666 NUM 3  ... or
;1481    1458  2     !   SET/DEFINE GROUP FRED ALL
;1482    1459  2     !   SET/DEFINE GROUP FRED KNOWN DTES ALL
;1483    1460  2     !
;1484    1461  3     BEGIN
;1485    1462  3     IF .NML$GB_ENTITY_FORMAT GTR 0 THEN
;1486    1463  4         BEGIN
;1487    1464  4         !
;1488    1465  4         ! SET/DEFINE X25-PROTOCOL GROUP FRED KNOWN DTES ALL - is illegal.
;1489    1466  4         !
;1490    1467  4         IF .NML$GL_PRS_FLGS [NML$V_PRS_ALL] AND
;1491    1468  5            (.NML$GL_PRS_FLGS [NML$V_PRS_QUALIFIER] AND
```

```
;  1492          1469  4              .NML$GB_QUALIFIER_FORMAT LEQ 0) THEN
;  1493          1470  4                  RETURN NML$_STS_IDE;
;  1494          1471  3          END;
;  1495          1472  2      END;
;  1496          1473  2
;  1497          1474  2  RETURN NML$_STS_SUC;
;  1498          1475  2
;  1499          1476  1  END;                              ! End of NML$PRM_QUAL_FORMAT



                                                  000C 00000          .ENTRY  NML$PRM_QUAL_FORMAT, Save R2,R3                 ; 1413
                                      53 00000000G  00  9E 00002      MOVAB   NML$GB_QUALIFIER_FORMAT, R3                     ;
                                      52 00000000G  00  9E 00009      MOVAB   NML$GL_PRS_FLGS, R2                             ;
                                      50 00000000G  00  9A 00010      MOVZBL  NML$GB_ENTITY_FORMAT, R0                        ; 1446
                                                    16  14 00017      BGTR    1$                                              ;
                                         00000000G  00  95 00019      TSTB    NML$GB_OPTIONS                                  ; 1448
                                                    1A  19 0001F      BLSS    2$                                              ;
                          16            62          01  E1 00021      BBC     #1, NML$GL_PRS_FLGS, 2$                         ; 1449
                          16            62          02  E1 00025      BBC     #2, NML$GL_PRS_FLGS, 3$                         ; 1450
                                                    63  95 00029      TSTB    NML$GB_QUALIFIER_FORMAT                         ; 1451
                                                    12  13 0002B      BEQL    3$                                              ;
                                                    0C  11 0002D      BRB     2$                                              ; 1452
                          0C            62          01  E1 0002F  1$: BBC     #1, NML$GL_PRS_FLGS, 3$                         ; 1467
                          08            62          02  E1 00033      BBC     #2, NML$GL_PRS_FLGS, 3$                         ; 1468
                                                    63  95 00037      TSTB    NML$GB_QUALIFIER_FORMAT                         ; 1469
                                                    04  12 00039      BNEQ    3$                                              ;
                                      50            12  CE 0003B  2$: MNEGL   #18, R0                                         ; 1470
                                                        04 0003E      RET                                                     ;
                                      50            01  D0 0003F  3$: MOVL    #1, R0                                          ; 1474
                                                        04 00042      RET                                                     ; 1476

; Routine Size:  67 bytes,    Routine Base:  $CODE$ + 040D
```

```
;  1501          1477  1 %SBTTL 'NML$PRM_CHANNELS  Check parameter value'
;  1502          1478  1 GLOBAL ROUTINE NML$PRM_CHANNELS =
;  1503          1479  1
;  1504          1480  1 !++
;  1505          1481  1 ! FUNCTIONAL DESCRIPTION:
;  1506          1482  1 !
;  1507          1483  1 ! The X-25 Protocol DTE parameter, channels, is passed by NCP in the form of
;  1508          1484  1 ! parameter id, channel pair, parameter id, channel pair...
;  1509          1485  1 ! It must be converted to a string.  Accumulate the string here.
;  1510          1486  1 !
;  1511          1487  1 !
;  1512          1488  1 ! FORMAL PARAMETERS:
;  1513          1489  1 !
;  1514          1490  1 !       NONE
;  1515          1491  1 !
;  1516          1492  1 ! IMPLICIT INPUTS:
;  1517          1493  1 !
;  1518          1494  1 !       NPARSE_BLOCK (pointed to by AP) contains the parsed parameter data.
;  1519          1495  1 !           NPA$L_FLDCNT is the parameter length.
;  1520          1496  1 !           NPA$L_FLDPTR is a pointer to the parameter in the received
;  1521          1497  1 !               message buffer.
;  1522          1498  1 !       NML$GL_PRMCODE contains the index into the change parameter table
;  1523          1499  1 !           (NML$AB_CPTABLE).
;  1524          1500  1 !
;  1525          1501  1 ! IMPLICIT OUTPUTS:
;  1526          1502  1 !
;  1527          1503  1 !       If the parameter is valid then a descriptor entry will be created for
;  1528          1504  1 !       it in NML$AW_PRM_DES and NML$GW_PRMDESCNT will be incremented.
;  1529          1505  1 !
;  1530          1506  1 ! ROUTINE VALUE:
;  1531          1507  1 ! COMPLETION CODES:
;  1532          1508  1 !
;  1533          1509  1 !       Always returns NML$_STS_SUC.
;  1534          1510  1 !
;  1535          1511  1 ! SIDE EFFECTS:
;  1536          1512  1 !
;  1537          1513  1 !--
;  1538          1514  1
;  1539          1515  2 BEGIN
;  1540          1516  2
;  1541          1517  2 $NPA_ARGDEF;                        ! Define NPARSE block reference
;  1542          1518  2
;  1543          1519  2 LITERAL CHAN_STR_SIZE = 80;
;  1544          1520  2
;  1545          1521  2 OWN
;  1546          1522  2     INDEX,                  ! Parameter stack index for channels entry.
;  1547          1523  2     CHANNEL_STRING: BBLOCK [CHAN_STR_SIZE];
;  1548          1524  2
;  1549          1525  2 LOCAL
;  1550          1526  2     COUNT;
;  1551          1527  2
;  1552          1528  2 IF NOT .NML$GL_PRS_FLGS [NML$V_PRS_CHANNELS] THEN
;  1553          1529  2     !
;  1554          1530  2     ! This is the first channel range in the NICE SET command.  Set
;  1555          1531  2     ! up the parameter descriptor list to accumulate all the channel
;  1556          1532  2     ! ranges in a string which is how it must be passed to the PSI
;  1557          1533  2     ! ACP.
```

```
;  1558          1534  2      !
;  1559          1535  2      BEGIN
;  1560          1536  2      NML$GL_PRS_FLGS [NML$V_PRS_CHANNELS] = 1;
;  1561          1537  2      !
;  1562          1538  2      ! Save the index into the parameter descriptor stack in order to add
;  1563          1539  2      ! the length of later channel pairs.  Then add the descriptor entry
;  1564          1540  2      ! for channels to the stack.
;  1565          1541  2      !
;  1566          1542  2      INDEX = .NML$GW_PRMDESCNT;
;  1567          1543  2      NML$SAVEPARAM (.NML$GL_PRMCODE, 0, CHANNEL_STRING);
;  1568          1544  2      END;
;  1569          1545  2
;  1570          1546  2  !
;  1571          1547  2  ! Make sure the number of channel pairs in the NICE message isn't overflowing
;  1572          1548  2  ! the internal channel string buffer.
;  1573          1549  2  !
;  1574          1550  2  COUNT = .NML$AW_PRM_DES [.INDEX, PDB$W_COUNT];
;  1575          1551  2  IF .COUNT GEQ CHAN_STR_SIZE THEN
;  1576          1552  2      NML$ERROR_2 (NMA$C_STS_PLO,
;  1577          1553  2                   NMA$C_PCXP_CHN);
;  1578          1554  2
;  1579          1555  2  !
;  1580          1556  2  ! Move the channel pair to the end of the string of channel pairs.  Then
;  1581          1557  2  ! increment the string length in the parameter descriptor stack.
;  1582          1558  2  !
;  1583          1559  2  CHANNEL_STRING [.COUNT,0,32,0] = ..NPARSE_BLOCK [NPA$L_FLDPTR];
;  1584          1560  2  NML$AW_PRM_DES [.INDEX, PDB$W_COUNT] = .COUNT + .NPARSE_BLOCK [NPA$L_FLDCNT];
;  1585          1561  2
;  1586          1562  2  NML$GL_PRMCODE = 0;                        ! Reset parsing code
;  1587          1563  2
;  1588          1564  2  RETURN NML$_STS_SUC
;  1589          1565  2
;  1590          1566  1  END;                                ! End of NML$PRM_CHANNELS


                                                            .PSECT  $OWN$,NOEXE,2

                                                  00002     .BLKB   2
                                                  00004 INDEX:  .BLKB   4
                                                  00008 CHANNEL_STRING:
                                                            .BLKB   80


                                                            .PSECT  $CODE$,NOWRT,2

                                                    003C 00000      .ENTRY  NML$PRM_CHANNELS, Save R2,R3,R4,R5          ; 1478
                                55 00000000G    00   9E 00002      MOVAB   NML$AW_PRM_DES+2, R5                        ;
                                54 00000000G    00   9E 00009      MOVAB   NML$GL_PRMCODE, R4                          ;
                                53 00000000'    00   9E 00010      MOVAB   INDEX, R3                                   ;
                                1C 00000000G    00   E8 00017      BLBS    NML$GL_PRS_FLGS+1, 1$                       ; 1528
                    00000000G   00              01   88 0001E      BISB2   #1, NML$GL_PRS_FLGS+1                       ; 1536
                                63 00000000G    00   3C 00025      MOVZWL  NML$GW_PRMDESCNT, INDEX                     ; 1542
                                          04    A3   9F 0002C      PUSHAB  CHANNEL_STRING                              ; 1543
                                                7E   D4 0002F      CLRL    -(SP)                                       ;
                                                64   DD 00031      PUSHL   NML$GL_PRMCODE                              ;
```

```
                    00000000G  00            03  FB 00033           CALLS   #3, NML$SAVEPARAM                          ;
                               50            63  D0 0003A 1$:       MOVL    INDEX, R0                                 ; 1550
                                           6540  7F 0003D           PUSHAQ  NML$AW_PRM_DES+2[R0]                      ;
                               52            9E  3C 00040           MOVZWL  @(SP)+, COUNT                             ;
                    00000050   8F            52  D1 00043           CMPL    COUNT, #80                                ; 1551
                                              0F  19 0004A           BLSS    2$                                        ;
                               7E      046A   8F  3C 0004C           MOVZWL  #1130, -(SP)                              ; 1552
                               7E            17  CE 00051           MNEGL   #23, -(SP)                                ;
                    00000000G  00            02  FB 00054           CALLS   #2, NML$ERROR_2                           ;
                                         04 A342  9F 0005B 2$:       PUSHAB  CHANNEL_STRING[COUNT]                     ; 1559
                               9E        14   BC  D0 0005F           MOVL    @20(NPARSE_BLOCK), @(SP)+                 ;
                               50            63  D0 00063           MOVL    INDEX, R0                                 ; 1560
                                           6540  7F 00066           PUSHAQ  NML$AW_PRM_DES+2[R0]                      ;
                         9E                52   10  AC  A1 00069   ADDW3   16(NPARSE_BLOCK), COUNT, @(SP)+           ;
                                               64  D4 0006E           CLRL    NML$GL_PRMCODE                            ; 1562
                               50              01  D0 00070           MOVL    #1, R0                                    ; 1564
                                                   04 00073           RET                                               ; 1566

; Routine Size:  116 bytes,    Routine Base:  $CODE$ + 0450

; 1591          1567  1
; 1592          1568  1
; 1593          1569  1
; 1594          1570  1 END                                     ! End of module
; 1595          1571  1
; 1596          1572  0 ELUDOM
```

```
;                                  PSECT SUMMARY

;       Name                    Bytes                           Attributes

;  $CODE$                        1220  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $PLIT$                           3  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $OWN$                           88  NOVEC,  WRT,  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        39        11          27        00:00.1
;  _$255$DUA28:[SHRLIB]NMALIBRY.L32;1      887        17         1          47        00:00.2
;  _$255$DUA28:[SYSLIB]STARLET.L32;1      9776         2         0         581        00:02.2
```

```
;                                       COMMAND QUALIFIERS

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

; Size:          1220 code + 91 data bytes
; Run Time:         00:27.1
; Elapsed Time:     01:05.5
; Lines/CPU Min:     3476
; Lexemes/CPU-Min: 11938
; Memory Used:  103 pages
; Compilation Complete
```

NML_PURGE
LIS

NML_PARINI
LIS

NML_NODFIL
LIS

NML_READ
LIS

NML_PARPRM
LIS

NML_MANIP
LIS