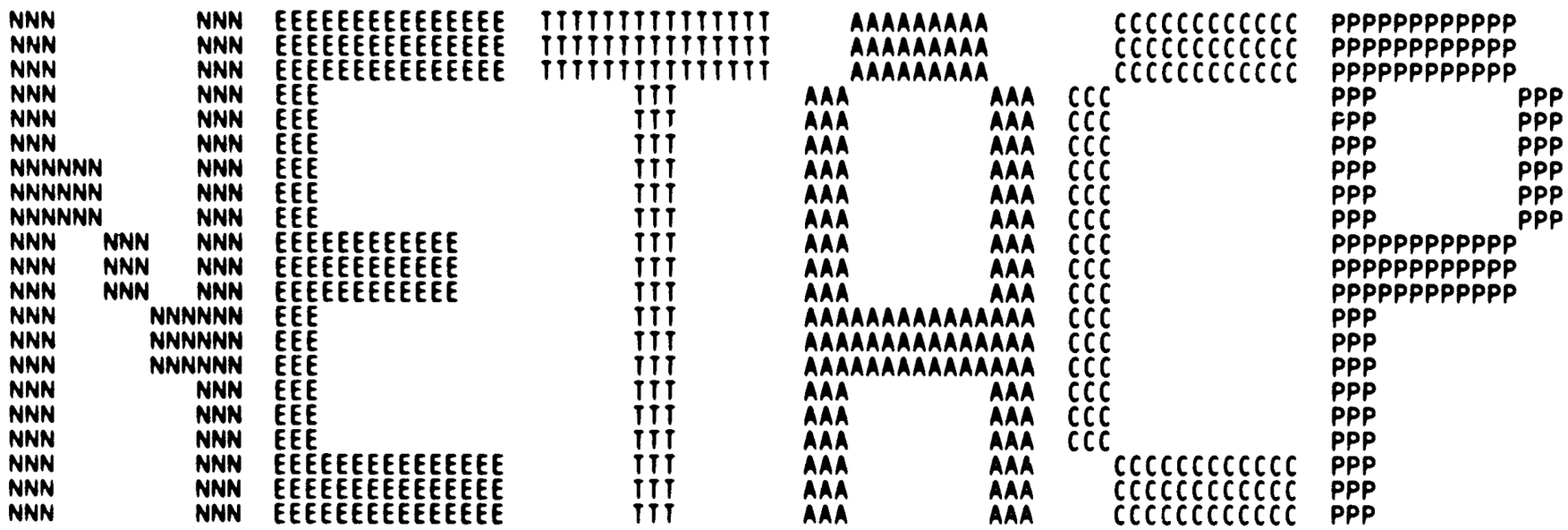
NNN NNN EEEEEEEEEEEEEEE TTTTTTTTTTTTTTT AAAAAAAAA CCCCCCCCCCC PPPPPPPPPPPP
NNN NNN EEEEEEEEEEEEEEE TTTTTTTTTTTTTTT AAAAAAAAA CCCCCCCCCCC PPPPPPPPPPPP
NNN NNN EEEEEEEEEEEEEEE TTTTTTTTTTTTTTT AAAAAAAAA CCCCCCCCCCC PPPPPPPPPPPP
NNN NNN EEE TTT AAA AAA CCC PPP PPP
NNN NNN EEE TTT AAA AAA CCC PPP PPP
NNN NNN EEE TTT AAA AAA CCC PPP PPP
NNNNNN NNN EEE TTT AAA AAA CCC PPP PPP
NNNNNN NNN EEE TTT AAA AAA CCC PPP PPP
NNNNNN NNN EEE TTT AAA AAA CCC PPP PPP
NNN NNN NNN EEEEEEEEEEE TTT AAA AAA CCC PPPPPPPPPPPP
NNN NNN NNN EEEEEEEEEEE TTT AAA AAA CCC PPPPPPPPPPPP
NNN NNN NNN EEEEEEEEEEE TTT AAA AAA CCC PPPPPPPPPPPP
NNN NNNNNN EEE TTT AAAAAAAAAAAAAAA CCC PPP
NNN NNNNNN EEE TTT AAAAAAAAAAAAAAA CCC PPP
NNN NNNNNN EEE TTT AAAAAAAAAAAAAAA CCC PPP
NNN NNN EEE TTT AAA AAA CCC PPP
NNN NNN EEE TTT AAA AAA CCC PPP
NNN NNN EEE TTT AAA AAA CCC PPP
NNN NNN EEEEEEEEEEEEEEE TTT AAA AAA CCCCCCCCCCC PPP
NNN NNN EEEEEEEEEEEEEEE TTT AAA AAA CCCCCCCCCCC PPP
NNN NNN EEEEEEEEEEEEEEE TTT AAA AAA CCCCCCCCCCC PPP

FILEID**NETCONECT

NE
VO

```
NN      NN  EEEEEEEEEE  TTTTTTTTTT    CCCCCCCC    OOOOOO    NN      NN  EEEEEEEEEE    CCCCCCCC  TTTTTTTTTT
NN      NN  EEEEEEEEEE  TTTTTTTTTT    CCCCCCCC    OOOOOO    NN      NN  EEEEEEEEEE    CCCCCCCC  TTTTTTTTTT
NN      NN  EE              TT      CC          OO      OO  NN      NN  EE          CC              TT
NN      NN  EE              TT      CC          OO      OO  NN      NN  EE          CC              TT
NNNN    NN  EE              TT      CC          OO      OO  NNNN    NN  EE          CC              TT
NN NN   NN  EE              TT      CC          OO      OO  NNNN    NN  EE          CC              TT
NN  NN  NN  EEEEEEEE        TT      CC          OO      OO  NN  NN  NN  EEEEEEEE    CC              TT
NN  NN  NN  EEEEEEEE        TT      CC          OO      OO  NN  NN  NN  EEEEEEEE    CC              TT
NN    NNNN  EE              TT      CC          OO      OO  NN    NNNN  EE          CC              TT
NN    NNNN  EE              TT      CC          OO      OO  NN    NNNN  EE          CC              TT
NN      NN  EE              TT      CC          OO      OO  NN      NN  EE          CC              TT      ....
NN      NN  EE              TT      CC          OO      OO  NN      NN  EE          CC              TT      ....
NN      NN  EEEEEEEEEE      TT        CCCCCCCC    OOOOOO    NN      NN  EEEEEEEEEE    CCCCCCCC        TT      ....
NN      NN  EEEEEEEEEE      TT        CCCCCCCC    OOOOOO    NN      NN  EEEEEEEEEE    CCCCCCCC        TT      ....

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
```

Table of contents

```
0000     1          .TITLE  NETCONECT - Process user connect requests
0000     2          .IDENT  'V04-000'
00C0     3          .DEFAULT DISPLACEMENT,LONG
0000     4
0000     5  ;*****************************************************************************
0000     6  ;*                                                                           *
0000     7  ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     8  ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     9  ;*  ALL RIGHTS RESERVED.                                                     *
0000    10  ;*                                                                           *
0000    11  ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
0000    12  ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
0000    13  ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
0000    14  ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
0000    15  ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
0000    16  ;*  TRANSFERRED.                                                             *
0000    17  ;*                                                                           *
0000    18  ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
0000    19  ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
0000    20  ;*  CORPORATION.                                                             *
0000    21  ;*                                                                           *
0000    22  ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
0000    23  ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    24  ;*                                                                           *
0000    25  ;*                                                                           *
0000    26  ;*****************************************************************************
0000    27  ;
0000    28  ; FACILITY:     NETWORK ACP
0000    29  ;
0000    30  ; ABSTRACT:
0000    31  ;
0000    32  ; This module performs processing for logical-link connect requests including
0000    33  ; connect initialize, connect confirm, and connect reject.
0000    34  ;
0000    35  ; The Network Connect Block (NCB) is parsed and an Internal Connect Block (ICB)
0000    36  ; containing the parse, is built and hung onto the IRP.  The IRP is requeued
0000    37  ; to the  NETDRIVER.
0000    38  ;
0000    39  ; NCBs have the same form as the translation of the logical name  "SYS$NET"
0000    40  ; and is shown below.
0000    41  ;
0000    42  ;     node"access control info"::"object=taskname/linknumber+userdata"
0000    43  ;
0000    44  ;       'node'      may be specified either by name or number.
0000    45  ;       'object'    may be specified either by name or number.
0000    46  ;       'taskname'  is required if the object number is zero and is only
0000    47  ;                   allowed if the object number is zero.
0000    48  ;
0000    49  ;
0000    50  ; ENVIRONMENT:
0000    51  ;
0000    52  ;       MODE = KERNEL
0000    53  ;
0000    54  ; AUTHOR:       A.Eldridge,   CREATION DATE: 10-JUN-79
0000    55  ;
0000    56  ; MODIFIED BY:
0000    57  ;
```

```
0000     58 ;        V03-035 PRB0344         Paul Beck       27-Jul-1984  13:21
0000     59 ;                Fix truncation error.
0000     60 ;
0000     61 ;        V03-034 ADE0035         Alan D. Eldridge        25-Jun-1984
0000     62 ;                Don't override XWB creation/insertion error code with
0000     63 ;                "SS$_NOLINKS".
0000     64 ;
0000     65 ;        V03-033 PRB0316         Paul Beck       8-Mar-1984  17:13
0000     66 ;                Resequence local symbols in PRS_NODE.
0000     67 ;                Allow endnode to offer larger buffer size if the buffer
0000     68 ;                associated with the line is larger than the executor buffer
0000     69 ;                size. This requires that the link be nonadaptive, but offers
0000     70 ;                performance wins.
0000     71 ;
0000     72 ;        V03-032 ADE0034         Alan D. Eldirdge        15-Feb-1984
0000     73 ;                Modify it use LLI database and insert XWB's into the LTB
0000     74 ;                vector. Send the External PID format in format type 2
0000     75 ;                connect requests.
0000     76 ;
0000     77 ;        V03-031 PRB0309         Paul Beck       23-Jan-1984  14:30
0000     78 ;                Do not make link nonadaptive if line buffer size equals
0000     79 ;                the executor buffer size. Undoes part of TMH0030.
0000     80 ;
0000     81 ;        V030    TMH0030         Tim Halvorsen   10-Jul-1983
0000     82 ;                Fix detection of "1 hop away" for purposes of using
0000     83 ;                line buffer size.  The previous check never worked
0000     84 ;                and always used the line buffer size if specified.
0000     85 ;                Allow normal NDI entries to specify an explicit output
0000     86 ;                circuit, overriding the decision algorithm.  This is
0000     87 ;                similar to loop nodes, but applies to nodes with real
0000     88 ;                remote addresses.
0000     89 ;                Remove check which ignored LINE BUFFER SIZE if it was
0000     90 ;                lower than the executor buffer size, so that as long
0000     91 ;                as the line buffer size parameter was explicitly specified,
0000     92 ;                is is used.
0000     93 ;
0000     94 ;        V029    TMH0029         Tim Halvorsen   31-May-1983
0000     95 ;                Fix problem with NODE ACCESS checking if the user
0000     96 ;                specified a node address without an area number.
0000     97 ;
0000     98 ;        V028    RNG0028         Rod Gamache     20-Apr-1983
0000     99 ;                Fix branch destination out of range.
0000    100 ;
0000    101 ;        V027    TMH0027         Tim Halvorsen   05-Mar-1983
0000    102 ;                Remove obsolete DLE code (replaced by completely
0000    103 ;                rewritten DLE module).
0000    104 ;
0000    105 ;        V026    TMH0026         Tim Halvorsen   14-Feb-1983
0000    106 ;                Remove node proxy access parameter.
0000    107 ;                Add support for "line buffer size" which can be used by a
0000    108 ;                system manager to override the executor buffer size on
0000    109 ;                a per-line basis.  This parameter has special meaning,
0000    110 ;                in that when used to increase the line's buffer size
0000    111 ;                higher than the executor buffer size, then all logical
0000    112 ;                links to adjacent nodes over this line become "non-adaptive",
0000    113 ;                and use the larger buffer size for optimized performance.
0000    114 ;                Add support for EPIDs.
```

```
0000    115 ;
0000    116 ;       V025    TMH0025         Tim Halvorsen   28-Dec-1982
0000    117 ;               Send username rather than PID with outgoing connects
0000    118 ;               if default access control is supplied to the remote node
0000    119 ;               (except the nonprivileged local node case).
0000    120 ;               Fix local outgoing connect case so that nonprivileged
0000    121 ;               access is supplied on the inbound side, not the outbound
0000    122 ;               side.  This fixes a problem with proxy that prevented
0000    123 ;               proxy from working on local connects unless the local NDI
0000    124 ;               proxy was set.
0000    125 ;               Fix long-standing bug which prevented outgoing default
0000    126 ;               access control from being applied because some junk in
0000    127 ;               the upper word of the NDI search key wasn't being zeroed.
0000    128 ;               This fixes both outgoing default access control for remote
0000    129 ;               nodes, and it fixes the privileged access control mechanism.
0000    130 ;               It also fixes loop nodes, which were failing to associate
0000    131 ;               the link with the proper circuit, and were using the local
0000    132 ;               LPD instead.
0000    133 ;               Fix loop node connect, so that if the circuit exists, but
0000    134 ;               has no LPD (the state is off), then an error is returned.
0000    135 ;
0000    136 ;       V024    TMH0024         Tim Halvorsen   29-Oct-1982
0000    137 ;               Add area routing support.
0000    138 ;               Fix DLE so that it matches by user channel as well
0000    139 ;               as PID, so that a cancel on another NET channel doesn't
0000    140 ;               blow away DLE channels.
0000    141 ;
0000    142 ;       V023    TMH0023         Tim Halvorsen   29-Sep-1982
0000    143 ;               Avoid check which ensures that a node is reachable
0000    144 ;               at connect time if we are an endnode.
0000    145 ;
0000    146 ;       V022    TMH0022         Tim Halvorsen   02-Sep-1982
0000    147 ;               Remove check of XWB state in DLE cancel routine.
0000    148 ;
0000    149 ;       V021    TMH0021         Tim Halvorsen   22-Jul-1982
0000    150 ;               Modify call to TEST_REACH, to use adjacency symbols
0000    151 ;               to determine the type of partner node.
0000    152 ;
0000    153 ;       V020    TMH0020         Tim Halvorsen   29-Jun-1982
0000    154 ;               Add $DYNDEF definition.
0000    155 ;
0000    156 ;       V019    TMH0019         Tim Halvorsen   09-Apr-1982
0000    157 ;               Fix proxy access checking for inbound connect requests
0000    158 ;               of zero-numbered objects with the name in the NCB.
0000    159 ;               It didn't correctly look up the proxy access parameter
0000    160 ;               in the named OBI entry, but used the number proxy access
0000    161 ;               value instead.
0000    162 ;               Pick up address of utility buffer, rather than referencing
0000    163 ;               a statically defined location.
0000    164 ;
0000    165 ;       V018    TMH0018         Tim Halvorsen   05-Mar-1982
0000    166 ;               Mark ACP in "dismount" state when the mount count goes
0000    167 ;               to zero, to avoid a race between the final DLE XWB coming
0000    168 ;               back from NETDRIVER and a new ACCESS function coming in
0000    169 ;               from a user.  The "dismount" state will signal the EXEC
0000    170 ;               to reject the QIO request.
0000    171 ;
```

```
0000   172 ;        X02-17  ADE0033         A.Eldridge      25-Jan-1982
0000   173 ;                Disallow default outbound access control if connect uses
0000   174 ;                proxy login.
0000   175 ;
0000   176 ;        X02-16  ADE0032         A.Eldridge      18-Jan-1982
0000   177 ;                Require OPER priv on IO$_ACCESS for circuit "direct-access".
0000   178 ;
0000   179 ;        X02-15  ADE0031         A.Eldridge      18-Dec-1981
0000   180 ;                Enter remote object name as the RID field (remote i.d.)
0000   181 ;                when initiating outbound connects.
0000   182 ;
0000   183 ;        X02-14  ADE0030         A.Eldridge      30-Nov-1981
0000   184 ;                Added proxy login support.
0000   185 ;
0000   186 ;        X02-13  ADE0029         A.Eldridge      11-Nov-1981
0000   187 ;                Identify local process by username rather than PID in crder
0000   188 ;                to allow the implementation of proxy logins at the remote
0000   189 ;                side of the link.
0000   190 ;
0000   191 ;        X02-12
0000   192 ;        -X02-10 ADE0028         A.Eldridge      1-Nov-1981
0000   193 ;                Fix bugs in "direct-link access" code.
0000   194 ;
0000   195 ;        X02-09                  A.Eldridge      1-Oct-1981
0000   196 ;                Put in "direct-link access" interface.
0000   197 ;
0000   198 ;        X02-08                  A.Eldridge      1-Oct-1981
0000   199 ;                Permanent modification to optionally restrict logical link
0000   200 ;                access based upon the "access state" of the remote node and
0000   201 ;                the privilege of the 'ocal user.
0000   202 ;
0000   203 ;        X02-07                  A.Eldridge      1-APR-1981
0000   204 ;                Tempory modification to optionally restrict outbound access
0000   205 ;                to selected nodes by nonprivileged users.  This is for DECUS
0000   206 ;                and NCC demos.
0000   207 ;
0000   208 ;        V02-04                  A.Eldridge      11-NOV-1979
0000   209 ;                Modify for new node, object, and task data base
0000   210 ;
0000   211 ;        V02-03                  S.G.D.  11-JUN-1979
0000   212 ;                Modify for routing.
0000   213 ;        V02-02  SGD00007        S.G.D.  22-NOV-1978     13:10
0000   214 ;                Allow multiple spaces and tabs in access control info.
0000   215 ;
0000   216 ;&
0000   217 ;& need to fix bug which disallows a null destination name on connect confirm
```

```
                     0000   219 .SBTTL  DECLARATIONS
                     0000   220 ;
                     0000   221 ; INCLUDE FILES:
                     0000   222 ;
                     0000   223         $ABDDEF
                     0000   224         $DRDEF
                     0000   225         $DYNDEF
                     0000   226         $IRPDEF
                     0000   227         $PRVDEF
                     0000   228         $JPIDEF
                     0000   229
                     0000   230         $CNRDEF
                     0000   231         $CNFDEF
                     0000   232
                     0000   233         $NETSYMDEF
                     0000   234         $NETUPDDEF
                     0000   235         $NSPMSGDEF      ; DNA architecture definitions & message formats
                     0000   236
                     0000   237         $ICBDEF
                     0000   238         $LTBDEF
                     0000   239         $NMADEF
                     0000   240         $NFBDEF
                     0000   241         $RCBDEF
                     0000   242         $ADJDEF
                     0000   243         $LPDDEF
                     0000   244         $XWBDEF
                     0000   245
                     0000   246 ;
                     0000   247 ; MACROS:
                     0000   248 ;
                     0000   249 .MACRO FILL_INC  NUMCHARS,STARTCHAR,STARTPOS    ; Fill range with
                     0000   250                                                 ; increasing values
                     0000   251         .=.-256+STARTPOS                        ; Reposition PC
                     0000   252
                     0000   253                 C=STARTCHAR
                     0000   254         .REPT   NUMCHARS                        ; Loop for each char.
                     0000   255         .BYTE   C                               ; Store character
                     0000   256                 C=C+1                           ; Bump character
                     0000   257         .ENDR
                     0000   258
                     0000   259         .=.-NUMCHARS-STARTPOS+256               ; Restore PC
                     0000   260 .ENDM
                     0000   261
                     0000   262
                     0000   263 ;
                     0000   264 ; EQUATED SYMBOLS:
                     0000   265 ;
00000009             0000   266 TAB     = ^X<09>                ; ASCII for tab
00000020             0000   267 SPACE   = ^X<20>                ; ASCII for space
                     0000   268
                     0000   269 ;
                     0000   270 ; OWN STORAGE:
                     0000   271 ;
00000000             0000   272         .PSECT  NET_PURE,NOWRT,NOEXE,LONG
                     0000   273
                     0000   274
                     0000   275 PRV_TAB:                                ; Field i.d.'s for privilege access
```

```
                                0000   276
                                0000   277          .CNFFLD ndi,s,pus                  ; Privileged user field i.d.
                                0004   278          .CNFFLD ndi,s,ppw                  ; Privileged password field i.d.
                                0008   279          .CNFFLD ndi,s,pac                  ; Privileged accoutn field i.d.
                     00000000   000C   280          .LONG   0
                                0010   281
                                0010   282 NONPRV_TAB:                                 ; Field i.d.'s for nonprivileged access
                                0010   283
                                0010   284          .CNFFLD ndi,s,nus                  ; Nonpriv user field i.d.
                                0014   285          .CNFFLD ndi,s,npw                  ; Nonpriv password field i.d.
                                0018   286          .CNFFLD ndi,s,nac                  ; Nonpriv account field i.d.
                     00000000   001C   287          .LONG   0
                                0020   288
42 41 39 38 37 36 35 34 33 32 31 30 0020   289 BIN_HEXASC:      .ASCII  /0123456789ABCDEF/ ; For binary to Hex Ascii conversion
                  46 45 44 43   002C
                                0030   290
                                0030   291 NET$AB_UPASCNUM::                           ; Translation table for upper
                                0030   292                                            ; case ASCII and numerics
00'00'00'00'00'00'00'00'00'00'00'00' 0030   293                  .BYTE   0[256]        ; Fill initially with terminator
00'00'00'00'00'00'00'00'00'00'00'00' 003C
00'00'00'00'00'00'00'00'00'00'00'00' 0048
00'00'00'00'00'00'00'00'00'00'00'00' 0054
00'00'00'00'00'00'00'00'00'00'00'00' 0060
00'00'00'00'00'00'00'00'00'00'00'00' 006C
00'00'00'00'00'00'00'00'00'00'00'00' 0078
00'00'00'00'00'00'00'00'00'00'00'00' 0084
00'00'00'00'00'00'00'00'00'00'00'00' 0090
00'00'00'00'00'00'00'00'00'00'00'00' 009C
00'00'00'00'00'00'00'00'00'00'00'00' 00A8
00'00'00'00'00'00'00'00'00'00'00'00' 00B4
00'00'00'00'00'00'00'00'00'00'00'00' 00C0
00'00'00'00'00'00'00'00'00'00'00'00' 00CC
00'00'00'00'00'00'00'00'00'00'00'00' 00D8
00'00'00'00'00'00'00'00'00'00'00'00' 00E4
00'00'00'00'00'00'00'00'00'00'00'00' 00F0
00'00'00'00'00'00'00'00'00'00'00'00' 00FC
00'00'00'00'00'00'00'00'00'00'00'00' 0108
00'00'00'00'00'00'00'00'00'00'00'00' 0114
00'00'00'00'00'00'00'00'00'00'00'00' 0120
                  00'00'00'00   012C
                                0130   294      FILL_INC 10,<^A"0">,<^A"0">          ; All numerics trans to themselves
                                0130   295      FILL_INC 26,<^A"A">,<^A"A">          ; All upper case "   "
                                0130   296      FILL_INC 26,<^A"A">,<^A"a">          ; All lower case "   " uppercase
                                0130   297
                                0130   298 NET$AB_OBJTRAN:                           ; Translation table for object names
                                0130   299          .REPT   256
                                C130   300          .BYTE   .-NET$AB_OBJTRAN         ; Fill up translation tabel with .=.
                          00    0130   301          .ENDR
                                0230   302      FILL_INC  1,<0>,<^A/"/>              ; Make /"/ a delimiter
                                0230   303      FILL_INC  1,<0>,<^A"/">              ; Make "/" a delimiter
                                0230   304      FILL_INC 26,<^A"A">,<^A"a">          ; All lower case "   " uppercase
                                0230   305
                     00000330   0230   306 NET$AB_ACC_TAB: .BLKB   256               ; Translation table for access control
                                0330   307
                                0330   308      FILL_INC 256,0,0                     ; Init so that each character translates
                                0330   309                                           ; to itself
                                0330   310      FILL_INC 1,0,SPACE                   ; Space is a terminator
```

```
                                                N 5

                     0330   311     FILL_INC 1,0,TAB                ; Tab   is a terminator
                     0330   312     FILL_INC 1,0,<^A/''/>           ; Quote is a terminator
                     0330   313
                     0330   314
                00000000    315         .PSECT  NET_IMPURE,WRT,NOEXE,LONG
                     0000   316
            00000000 0000   317 ACC_TAB:        .LONG   0           ; Points to current access table
            00000000 0004   318 NDI_PTR:        .LONG   0           ; Points to NDI describing the node
            00000000 0008   319 OBI_PTR:        .LONG   0           ; Points to destination OBI
                     000C   320
00000000 00000000    000C   321 OBJ_Q_DESC:     .QUAD   0           ; Object specifier from NCB
00000000 00000000    0014   322 TSK_Q_DESC:     .QUAD   0           ; Task   specifier from NCB
                     001C   323
                  00 001C   324 NDI_B_ACC:      .BYTE   0           ; NDI access state
                  00 001D   325 OBI_B_PRX:      .BYTE   0           ; OBI proxy access state
                  00 001E   326 INT_B_PRX:      .BYTE   0           ; Internal proxy access state
            00000020 001F   327                 .BLKB   1           ; (spare for alignment)
                     0020   328
00000000 00000000    0020   329 JPI_Q_IOSB:     .QUAD   0           ; IOSB for GET_JPI
            00000000 0028   330 JPI_B_UNAME:    .LONG   0           ; Returns resultant user name length
            00000038 002C   331 JPI_T_UNAME:    .BLKB   12          ; Returns user name
                     0038   332
                     0038   333 JPI_ITEM_LIST:                      ; $GETJPI item list for logical links
                 000C 0038   334                 .WORD   12          ; Size of username buffer
                 0202 003A   335                 .WORD   JPI$_USERNAME ; I.d. of username parameter
            0000002C' 003C   336                 .LONG   JPI_T_UNAME ; Address of username buffer
            00000028' 0040   337                 .LONG   JPI_B_UNAME ; Address of buffer to return length
            00000000  0044   338                 .LONG   0           ; Terminate the list
                      0048   339
                00000000    340         .PSECT  NET_CODE,NOWRT,EXE
                     0000   341
```

```
                     0000   343 .SBTTL  NET$CONNECT     - IO$_ACCESS $QIO Procesing
                     0000   344 ;++
                     0000   345 ;
                     0000   346 ; This routine processes user connect inits or confirms.  Parameters and
                     0000   347 ; connect block (NCB) are validated.  Information in the NCB is passed to
                     0000   348 ; NETDRIVER in an ICB (Internal Connect Block).
                     0000   349 ;
                     0000   350 ; Connect Initiates and Confirms are distinguished by the value of the
                     0000   351 ; word following the remote process identifier:
                     0000   352 ;
                     0000   353 ;       Connect Initiates use a 0
                     0000   354 ;       Connect Confirms  use the supplied value (i.e, the local link number)
                     0000   355 ;
                     0000   356 ;
                     0000   357 ; INPUTS:       R5      Logical-link UCB address
                     0000   358 ;               R3      IRP address
                     0000   359 ;--
                     0000   360 NET$CONNECT::                                ; Parse NCB
                0000 0000   361         .WORD   0                           ; Entry
                     0002   362
00000004'EF    D4   0002   363         CLRL    NDI_PTR                     ; No NDI pointer yet
00000008'EF    D4   0008   364         CLRL    OBI_PTR                     ; No OBI pointer yet
          56   D4   000E   365         CLRL    R6                          ; No ICB yet
                     0010   366         ;
                     0010   367         ;   Get the Network Connect Block (NCB) descriptor
                     0010   368         ;
 54   2C B3    D0   0010   369         MOVL    @IRP$L_SVAPTE(R3),R4                            ; ABD ptr
 55   12 A4    3C   0014   370         MOVZWL  <ABD$C_LENGTH*ABD$C_NAME>+ABD$W_COUNT(R4),R5    ; NCB lth
 50   10 A4    9E   0018   371         MOVAB   <ABD$C_LENGTH*ABD$C_NAME>+ABD$W_TEXT(R4),R0     ; Offset
                     001C   372                                                                ; to text
    54   80    9E   001C   373         MOVAB   (R0)+,R4                    ; Copy the address and add 1
                     001F   374                                             ; for access mode field
         51   64    3C   001F   375         MOVZWL  (R4),R1                 ; Get offset to text
    54   50   51    C1   0022   376         ADDL3   R1,R0,R4                ; Point to device name string
         58   54    D0   0026   377         MOVL    R4,R8                   ; Copy name address
         57   55    D0   0029   378         MOVL    R5,R7                   ; Copy name size
         55   54    C0   002C   379         ADDL    R4,R5                   ; Point R5 past last NCB byte
                     002F   380         ;
                     002F   381         ;   Allocate an Internal Connect Block (ICB) to hold the parse
                     002F   382         ;   of the NCB.
                     002F   383         ;
  51   A3 8F    9A   002F   384         MOVZBL  #ICB$C_LENGTH,R1                ; Set block length
00000000'EF    16   0033   385         JSB     NET$ALONPGD_Z                   ; Allocate/zero from non-payed pool
       03 50    E8   0039   386         BLBS    R0,5$                           ; If error detected,
        0118    31   003C   387         BRW     ACCESS_DONE                     ; then exit with error status in R0
     56   52    D0   003F   388 5$:     MOVL    R2,R6                           ; Copy ICB pointer
                     0042   389         ;
                     0042   390         ;   Init ICB values obtained from LNI data base
                     0042   391         ;
                     0042   392         ASSUME  CNR$L_FLINK  EQ  0
50   00000000'EF    D0   0042   393         MOVL    NET$GL_PTR_VCB,R0                       ; Point at the RCB
   04 A6    78 A0    B0   0049   394         MOVW    RCB$W_TIM_CNO(R0),ICB$W_TIM_OCON(R6)    ; Outbound connect timer
   06 A6    74 A0    B0   004E   395         MOVW    RCB$W_TIM_IAT(R0),ICB$W_TIM_INACT(R6)   ; Inactivity timer
   0C A6    63 A0    9B   0053   396         MOVZBW  RCB$B_ECL_RFA(R0),ICB$W_RETRAN(R6)      ; Max rexmission count
   0E A6    64 A0    9B   0058   397         MOVZBW  RCB$B_ECL_DFA(R0),ICB$W_DLY_FACT(R6)    ; Rexmt delay factor
   10 A6    65 A0    9B   005D   398         MOVZBW  RCB$B_ECL_DWE(R0),ICB$W_DLY_WGHT(R6)    ; Rexmt delay weight
   12 A6    7C A0    B0   0062   399         MOVW    RCB$W_ECLSEGSIZ(R0),ICB$W_SEGSIZ(R6)    ; Segment size
```

```
                                0067   400          ;
                                0067   401          ;
                                0067   402          ;   Enter the local process name using NSP format type 1 and the PID
                                0067   403          ;   converted to ascii as the counted string.  This is the default
                                0067   404          ;   which may be overridded below, and is compatible with earlier
                                0067   405          ;   releases.
                                0067   406          ;
    50   00000000'EF    D0      0067   407          MOVL    NET$GL_PTR_VCB,R0       ; Point to RCB
               66 A0    90      006E   408          MOVB    RCB$B_ECL_DAC(R0),-     ;
         0000001C'EF             0071   409                  NDI_B_ACC               ; Setup default NDI access
               67 A0    90      0076   410          MOVB    RCB$B_ECL_DPX(R0),-     ;
         0000001D'EF             0079   411                  OBI_B_PRX               ; Setup default OBI proxy access
                  03    90      007E   412          MOVB    #NMA$C_ACES_BOTH,-      ; Setup default internal proxy access
         0000001E'EF             0080   413                  INT_B_PRX               ;
    52   00000000'EF    D0      0085   414          MOVL    NET$GL_SAVE_IRP,R2      ; Get current IRP
          4C A2    56    D0      008C   415          MOVL    R6,IRP$L_DIAGBUF(R2)    ; Save ICB for NETDRIVER
          50   0C A2    D0      0090   416          MOVL    IRP$L_PID(R2),R0        ; Get users PID
             58   08    D0      0094   417          MOVL    #8,R8                   ; Convert it to 8 ascii chars
          57   14 A6    9E      0097   418          MOVAB   ICB$B_LPRNAM(R6),R7     ; Get output pointer
               87   0B    90      009B   419          MOVB    #11,(R7)+               ; Total size including counted
                                009E   420                                          ; ascii PID, object and format type
               87   01    B0      009E   421          MOVW    #1,(R7)+                ; Format type 1, object type 0
               87   08    90      00A1   422          MOVB    #8,(R7)+                ; Setup count field for PID
               57   08    C0      00A4   423          ADDL    #8,R7                   ; Point R7 past end of dst field
                    51    D4      00A7   424          CLRL    R1                      ; Clear high order dividend
    52   50     50   10    7B      00A9   425  10$:    EDIV    #16,R0,R0,R2            ; Divide by 16, get remainder
         77   0020'C2    90      00AE   426          MOVB    BIN_HEXASC(R2),-(R7)    ; Convert to ASCII and store
                 F3 58    F5      00B3   427          SOBGTR  R8,10$                  ; Loop for 8 characters
                                00B6   428          ;
                                00B6   429          ;
                                00B6   430          ;   Parse the NCB
                                00B6   431          ;
                  0109    30    00B6   432          BSBW    PRS_NCB                 ; Parse the NCB
                06 50     E9    00B9   433          BLBC    R0,20$                  ; If LBC then error
                 0524     30    00BC   434          BSBW    CHECK_ACCESS            ; See if connect is allowed to node
                03 50     E8    00BF   435          BLBS    R0,30$                  ; If LBS then yes
                 0092     31    00C2   436  20$:    BRW     ACCESS_DONE             ; Exit
                                00C5   437  30$:    ;
                                00C5   438          ;
                                00C5   439          ;  If outbound proxy logins are allowed then identify the local
                                00C5   440          ;  process via format type 2 with the binary PID in the 'UIC' field
                                00C5   441          ;  and the username as the 12 byte counted string.
                                00C5   442          ;
                                00C5   443          $DISPATCH  TYPE=B,OBI_B_PRX -   ; Goto ACCESS_DONE if proxy disallowed
                                00C5   444          <-
                                00C5   445            <NMA$C_ACES_INCO, ACCESS_DONE>-
                                00C5   446            <NMA$C_ACES_NONE, ACCESS_DONE>-
                                00C5   447          >
                                00D1   448          $DISPATCH  TYPE=B,INT_B_PRX -   ; Goto ACCESS_DONE if proxy disallowed
                                00D1   449          <-
                                00D1   450            <NMA$C_ACES_INCO, ACCESS_DONE>-
                                00D1   451            <NMA$C_ACES_NONE, ACCESS_DONE>-
                                00D1   452          >
    52   00000000'EF    D0      00DD   453          MOVL    NET$GL_SAVE_IRP,R2      ; Get current IRP
          50   0C A2    D0      00E4   454          MOVL    IRP$L_PID(R2),R0        ; Get internal PID for process
         00000000'GF    16      00E8   455          JSB     G^EXE$IPID_TO_EPID      ; Convert to EPID format
               53   50    D0      00EE   456          MOVL    R0,R3                   ; Save EPID in R3
```

```
                     50    DD   00F1    457          PUSHL   R0                      ; Push EPID on stack
                50   5E    D0   00F3    458          MOVL    SP,R0                   ; Get address of EPID
                                00F6    459          $GETJPI_S -                     ;
                                00F6    460              PIDADR = (R0),-             ; EPID of process of interest
                                00F6    461              EFN    = #NET$C_EFN_WAIT,-  ; Event flag
                                00F6    462              IOSB   = JPI_Q_IOSB,-       ; IOSB
                                00F6    463              ITMLST = JPI_ITEM_LIST      ; Item list for return
                5E   04    C0   0111    464          ADDL    #4,SP                   ; Pop EPID off stack
                  40 50    E9   0114    465          BLBC    R0,ACCESS_DONE          ; Br on error
                                0117    466          $WAITFR_S  EFN = #NET$C_EFN_WAIT; Wait for $GETJPI to finish
          30 00000020'EF    E9   0120    467          BLBC    JPI_Q_IOSB,ACCESS_DONE  ; Br on error
       50   00000028'EF    9A   0127    468          MOVZBL  JPI_B_UNAME,R0          ; Get string size
                    24    13   012E    469          BEQL    40$                     ; If EQL then skip this
                0C  50    91   0130    470          CMPB    R0,#12                  ; Maximum name in NSP is 16
                    1F    1A   0133    471          BGTRU   40$                     ; If GTRU then out of range
             57   14 A6    9E   0135    472          MOVAB   ICB$B_LPRNAM(R6),R7     ; Get output pointer
          87   50    07    81   0139    473          ADDB3   #7,R0,(R7)+             ; Total size including username, PID,
                                013D    474                                          ; object type, and format type
                87  02    B0   013D    475          MOVW    #2,(R7)+                ; Format type 2, object type 0
                87  53    D0   0140    476          MOVL    R3,(R7)+                ; Enter binary EPID in "UIC" field
                87  50    90   0143    477          MOVB    R0,(R7)+                ; Setup count field for username
                7E  54    7D   0146    478          MOVQ    R4,-(SP)                ; Save NCB descriptor
    67  0000002C'EF  50    28   0149    479          MOVC3   R0,JPI_T_UNAME,(R7)     ; Move the username
                54  8E    7D   0151    480          MOVQ    (SP)+,R4                ; Restore NCB descriptor
                50  00'     D0 0154    481 40$:     MOVL    S^#SS$_NORMAL,R0        ; Setup status
                                0157    482                                          ;
                                0157    483 ACCESS_DONE:
       53   00000000'EF    D0   0157    484          MOVL    NET$GL_SAVE_IRP,R3      ; Recover IRP address
                  11 50    E8   015E    485          BLBS    R0,10$                  ; Br if successful
                4C A3  50   3C   0161    486          MOVZWL  R0,IRP$L_DIAGBUF(R3)    ; Save error code for NETDRIVER
                   50  56   D0   0165    487          MOVL    R6,R0                   ; Copy block address for deallocate
                       08   13   0168    488          BEQL    10$                     ; Br if none
             00000000'EF    16   016A    489          JSB     NET$DEALLOCATE          ; Deallocate the block
                       41   11   0170    490          BRB     100$                    ; Take common exit
                02 A6  B5   0172    491 10$:     TSTW    ICB$W_LOCLNK(R6)        ; Connect Initiate or Confirm ?
                   3C  12   0175    492          BNEQ    100$                    ; If NEQ, Confirm (or Reject)
                                0177    493          ;
                                0177    494          ;
                                0177    495          ;    A zero LOCLNK means that this is a Connect Initiate.  Allocate
                                0177    496          ;    an XWB and LLI and insert them in their respective databases.
                                0177    497          ;
                                0177    498          ;
                55   1C A3    D0   0177    499          MOVL    IRP$L_UCB(R3),R5        ; Get UCB address
                51   0C A3    D0   017B    500          MOVL    IRP$L_PID(R3),R1        ; Get PID
             53   008D C6     3C   017F    501          MOVZWL  ICB$W_REMNOD(R6),R3     ; Get remote node address
                50   07      D0   0184    502          MOVL    #NETUPD$_CRELNK,R0      ; Function code
             00000000'EF      16   0187    503          JSB     CALL_NETDRIVER          ; Tell Netdriver
                53   50      D0   018D    504          MOVL    R0,R3                   ; Get allocated XWB address
                     1F      18   0190    505          BGEQ    40$                     ; If GEQ, failed
                             0192    506                                          ;
                0048 8F      BB   0192    507          PUSHR   #^M<R3,R6>              ; Save XWB,ICB
             00000000'EF      16   0196    508          JSB     NET$PROC_XWB            ; Insert XWB, create LLI, etc.
                0048 8F      BA   019C    509          POPR    #^M<R3,R6>              ; Recover XWB,ICB
                  0E 50      E9   01A0    510          BLBC    R0,40$                  ; If LBC, XWB was deallocated
                  3E A3      B0   01A3    511          MOVW    XWB$W_LOCLNK(R3),-      ; Setup local link number
                  02 A6           01A6    512                  ICB$W_LOCLNK(R6)        ;
                     09      11   01A8    513          BRB     100$                    ; Tack common exit
```

```
                        01AA   514
                        01AA   515
50  00000000'8F   D0    01AA   515          MOVL    #SS$_NOLINKS,R0          ; Setup error code
                        01B1   516                                           ;& NO LONGER USED
              A4   11   01B1   517 40$:      BRB     ACCESS_DONE              ; Deal with the error
                        01B3   518                                           ;
53  00000000'EF   D0    01B3   519 100$:     MOVL    NET$GL_SAVE_IRP,R3       ; Recover IRP address
              20   A8    01BA   520          BISW    #NET$M_RQIRP,-           ;
      00000000'EF         01BC   521                  NET$GL_FLAGS             ; Give the IRP back to NETDRIVER
                  04    01C1   522          RET
                        01C2   523
```

```
                     01C2    525 .SBTTL  PRS_NCB - Parse Network Connect Block
                     01C2    526 ;+
                     01C2    527 ;
                     01C2    528 ; INPUTS:       R6      Ptr to the ICB
                     01C2    529 ;               R5      Ptr to first byte beyond the NDB
                     01C2    530 ;               R4      Ptr to first byte in the NDB
                     01C2    531 ;
                     01C2    532 ;               All other registers are scratch
                     01C2    533 ;
                     01C2    534 ; OUTPUTS:      R6      Preserved
                     01C2    535 ;               R0      Status code
                     01C2    536 ;
                     01C2    537 ;-
                     01C2    538 PRS_NCB:
          FE3B'  30  01C2    539         BSBW    NET$GETUTLBUF           ; Obtain use of the utility buf
     5F 8F    64  91  01C5    540         CMPB    (R4),#^A"_"             ; Is there a prefixed underscore?
              02  12  01C9    541         BNEQ    20$                     ; If NEQ no
              54  D6  01CB    542         INCL    R4                      ; Pass over it
              47  10  01CD    543 20$:    BSBB    PRS_NODE                ; Parse nodename, get NDI block
            3D 50  E9  01CF    544         BLBC    R0,TOO$                 ; Br if error
     3C A6    01  8E  01D2    545         MNEGB   #1,ICB$B_ACCESS(R6)     ; Flag 'no access control yet'
            0207  30  01D6    546         BSBW    PRS_ACCESS              ; Parse access control field
            33 50  E9  01D9    547         BLBC    R0,TOO$                 ; Br if error
     84   3A3A 8F  B1  01DC    548         CMPW    #^A"::",(R4)+           ; Correct delimiter
              2D  12  01E1    549         BNEQ    200$                    ; Br if not
            024D  30  01E3    550         BSBW    PRS_OBJECT              ; Parse the target object name
            26 50  E9  01E6    551         BLBC    R0,TOO$                 ; Br if error
            03A7  30  01E9    552         BSBW    PRS_END                 ; Parse remainder of the NCB
            20 50  E9  01EC    553         BLBC    R0,TOO$                 ; Br if error
     3C A6    FF 8F  91  01EF    554         CMPB    #-1,ICB$B_ACCESS(R6)    ; Any access control yet ?
              06  12  01F4    555         BNEQ    50$                     ; If NEQ then yes
            0456  30  01F6    556         BSBW    DFLT_ACCESS             ; Use the default
            13 50  E9  01F9    557         BLBC    R0,100$                 ; Br if error
          008D C6  B5  01FC    558 50$:    TSTW    ICB$W_REMNOD(R6)        ; Address = 0 ?
                OD  12  0200    559         BNEQ    100$                    ; If not, branch
                    0202    560                                             ; Else use the local address
     51 00000000'EF  D0  0202    561         MOVL    NET$GL_PTR_VCB,R1       ; Get RCB
              0E A1  B0  0209    562         MOVW    RCB$W_ADDR(R1),-        ;
          008D C6        020C    563                 ICB$W_REMNOD(R6)        ; Store local address
                    05  020F    564 100$:   RSB
                    0210    565
     50   0000'8F  3C  0210    566 200$:   MOVZWL  #SS$_IVDEVNAM,R0        ; Setup error code
                    05  0215    567         RSB                             ; Return error
```

```
                     0216   569 .SBTTL  PRS_NODE        - Parse NCB nodename
                     0216   570 ;+
                     0216   571 ;
                     0216   572 ; Parse the node identifier and find the appropriate NDI block.  If all
                     0216   573 ; numerics then convert from decimal to binary and use the NDI with the
                     0216   574 ; same address and null assoc. line (if not found then use null NDI).
                     0216   575 ;
                     0216   576 ; If the number is zero or the nodename is unspecified then treat as if
                     0216   577 ; the local nodename were used.  The local node number is always stored
                     0216   578 ; as a zero in all NDI blocks -- the actual local node number is found
                     0216   579 ; in the LNI block.
                     0216   580 ;
                     0216   581 ; The parse does not include the terminator which may be '' or ::
                     0216   582 ;
                     0216   583 ; INPUTS:       R6      Ptr to the ICB
                     0216   584 ;                R5      Ptr to first byte beyond the NDB
                     0216   585 ;                R4      Ptr to first byte in the NDB
                     0216   586 ;
                     0216   587 ;                All other are scratch
                     0216   588 ;
                     0216   589 ; OUTPUTS:      R6      Preserved
                     0216   590 ;                R5      Preserved
                     0216   591 ;                R4      Advance by bytes parsed
                     0216   592 ;                R0      Status code
                     0216   593 ;
                     0216   594 ;                ICB$W_REMNOD    Remote Node address -- 0 if its the local node
                     0216   595 ;                ICB$W_PATH      Path index of line to use to get to node.
                     0216   596 ;                NDI_PTR         Address of NDI CNF or 0 if none
                     0216   597 ;
                     0216   598 PRS_NODE:                                    ; Parse NCB nodename
           66   B4   0216   599         CLRW    ICB$W_PATH(R6)              ; Assume path zero
       59   06   9A   0218   600         MOVZBL  S^#NET$C_MAXNODNAM,R9       ; Indicate max size of nodename
58 00000000'EF   D0   021B   601         MOVL    NET$GL_UTLBUF,R8            ; Point to output buffer
         054D   30   0222   602         BSBW    GET_STR_NUM                 ; Returns:
                     0225   603                                             ; R8  name pointer
                     0225   604                                             ; R7  name string size
                     0225   605                                             ; R4  advanced by chars parsed
                     0225   606                                             ; R3  garbage
                     0225   607                                             ; R2  numeric value if LBS in R1
                     0225   608                                             ;     zero if null string
                     0225   609                                             ; R1  LBC if ascii string
                     0225   610                                             ;     LBS if numeric or null
                     0225   611                                             ; R0  garbage
                     0225   612
5B 00000000'EF   D0   0225   613         MOVL    NET$GL_CNR_NDI,R11          ; Setup root of NDI list
           5A   D4   022C   614         CLRL    R10                         ; Indicate no current NDI
         36 51   E9   022E   615         BLBC    R1,40$                      ; Br if Ascii nodename
       2E   64   91   0231   616         CMPB    (R4),#^A"."                 ; Is it of the form "area.node"?
            1A   12   0234   617         BNEQ    20$                         ; If not, use the number as the node
            54   D6   0236   618         INCL    R4                          ; Skip the delimiter
            52   DD   0238   619         PUSHL   R2                          ; Save area number
         0535   30   023A   620         BSBW    GET_STR_NUM                 ; Get the node number within area
            53 8ED0   023D   621         POPL    R3                          ; Restore area number
         08 51   E8   0240   622         BLBS    R1,10$                      ; If numeric, then it's ok
    50 0000'8F   3C   0243   623         MOVZWL  #SS$_IVDEVNAM,R0            ; Setup error code
         018D   31   0248   624         BRW     160$                        ; Report the error
       0A   53   F0   024B   625 10$:    INSV    R3,#TR4$V_ADDR_AREA,-       ; Combine area and node number
```

```
                52   06          024E   626                      #TR4$S_ADDR_AREA,R2
        59  00000000'EF   D0      0250   627 20$:     MOVL       NET$GL_PTR_VCB,R9          ; Get RCB
             0E A9    52   B1     0257   628          CMPW       R2,RCB$W_ADDR(R9)          ; Is this the local node?
                      02   12     025B   629          BNEQ       30$                        ; Br if address not local
                      52   D4     025D   630          CLRL       R2                         ; 0 is used to indicate the local node
                                  025F   631          ;
                                  025F   632          ;   The node has been specified by address in the NCB.  Attempt to find
                                  025F   633          ;   the associated NCB and continue.
                                  025F   634          ;
                58    52   D0     025F   635 30$:     MOVL       R2,R8                      ; Use as search value
                   FD9B'   30     0262   636          BSBW       NET$NDI_BY_ADD             ; Find the NDI with matching address
                      27   11     0265   637          BRB        60$                        ; R10 = NDI address, 0 if no match
                                  0267   638          ;
                                  0267   639          ;   The node has been specified by name in the NCB.   Find the NDI.  If
                                  0267   640          ;   its not there return an error since we cannot determine the node
                                  0267   641          ;   address.
                                  0267   642          ;
            50  0000'8F   3C      0267   643 40$:     MOVZWL     #SS$_NOSUCHNODE,R0         ; Establish error code
                                  026C   644          $SEARCH    egl,ndi,s,nna              ; Find the NDI block
                   03 50   E8     027B   645          BLBS       R0,50$                     ; If LBS then found
                    0157   31     027E   646          BRW        160$                       ; ...else return error
                                  0281   647 50$:     $GETFLD    ndi,l,add                  ; Get node address - its always there
                                  028E   648                                                ; and its value is 0 for the local node
                                  028E   649 60$:      ;
                                  028E   650          ;   At this point  R8  = node address (zero if local)
                                  028E   651          ;                  R10 = NDI block address (zero if none)
                                  028E   652          ;
                                  028E   653          ;   NOTE:  At this point, R8 may not be a "normalized" address,
                                  028E   654          ;   which means that if the area number was not specified, the
                                  028E   655          ;   homearea has not yet been defaulted!
                                  028E   656          ;
        008D C6      58   B0      028E   657          MOVW       R8,ICB$W_REMNOD(R6)        ; Store the address
  00000004'EF        5A   D0      0293   658          MOVL       R10,NDI_PTR                ; Save the NDI CNF pointer
                     17   13      029A   659          BEQL       70$                        ; If EQL then none
                                  029C   660          $GETFLD    ndi,l,acc                  ; Get access state
                07   50   E9      02A9   661          BLBC       R0,70$                     ; If LBC then not set
  0000001C'EF        58   90      02AC   662          MOVB       R8,NDI_B_ACC               ; Else override default
                                  02B3   663 70$:      ;
                                  02B3   664          ;   See if node is reachable
                                  02B3   665          ;
  51    00000000'EF   D0      02B3   666          MOVL       NET$GL_PTR_VCB,R1          ; Get RCB address
    52     008D C6      3C      02BA   667          MOVZWL     ICB$W_REMNOD(R6),R2        ; Get node address
                     5A   13      02BF   668          BEQL       100$                       ; If zero, then skip this
                     0A   EF      02C1   669          EXTZV      #TR4$V_ADDR_AREA,-         ; Get the remote area number
    50     52        06           02C3   670                     #TR4$S_ADDR_AREA,R2,R0
                     0B   12      02C6   671          BNEQ       80$                        ; If area = 0, then use our area
           008B C1   F0           02C8   672          INSV       RCB$B_HOMEAREA(R1),-       ; Always enforce our area set in
                     0A           02CC   673                     #TR4$V_ADDR_AREA,-         ; node addr, so that returning NSP
    008D C6          06           02CD   674                     #TR4$S_ADDR_AREA,ICB$W_REMNOD(R6) ; msgs match on node addr
                     07   11      02D1   675          BRB        90$                        ; Check node reachability
    008B C1          50   91      02D3   676 80$:     CMPB       R0,RCB$B_HOMEAREA(R1)      ; Our area?
                     41   12      02D8   677          BNEQ       100$                       ; If not, skip reachability check
    05     008A C1   91           02DA   678 90$:     CMPB       RCB$B_ETY(R1),#ADJ$C_PTY_PH4N ; Are we an endnode?
                     2A   12      02DF   679          BNEQ       95$                        ; If not, do reachability check
                                  02E1   680          ;
                                  02E1   681          ;   If the remote node is an endnode, there is only one adjacency
                                  02E1   682          ;   available. If that circuit has a buffer size larger than the
```

```
                          02E1   683          ;   executor buffer size, we can gain some throughput by making the
                          02E1   684          ;   link nonadaptive and offering to use the larger buffer size.
                          02E1   685          ;   However, we can only do this if we are certain that the target
                          02E1   686          ;   is one hop away. The only way to do this is to ask NETDRIVER to
                          02E1   687          ;   find it in the cache. If it's not in the cache, we don't know
                          02E1   688          ;   that it's one hop away, and we don't offer a big buffer.
                          02E1   689          ;
          0FFC 8F    BB   02E1   690          PUSHR   #^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ; Save registers
53  00000000'EF    D0   02E5   691          MOVL    NET$GL_SAVE_IRP,R3      ; Recover IRP address
 55     1C A3    D0   02EC   692          MOVL    IRP$L_OCB(R3),R5        ; Get network UCB address
    54     52    D0   02F0   693          MOVL    R2,R4                   ; Get node address of target
    52     51    D0   02F3   694          MOVL    R1,R2                   ; Copy RCB address
    50     0F    D0   02F6   695          MOVL    #NETUPD$_TEST_ADJ,R0    ; Function code
  00000000'EF    16   02F9   696          JSB     CALL_NETDRIVER          ; Tell Netdriver
       1F 50    E9   02FF   697          BLBC    R0,1T5$                 ; If LBC, target not in cache
                          0302   698          ;
                          0302   699          ; We have ascertained that the target node is one hop away.
                          0302   700          ; Join common code to decide whether to offer a larger buffer.
                          0302   701          ;
58    00AA C2    3C   0302   702          MOVZWL  RCB$W_DRT(R2),R8        ; Get ADJ index for designated router
          18    13   0307   703          BEQL    115$                    ; If EQL, none: don't bother
          2E    11   0309   704          BRB     125$                    ; Join common code
                          030B   705          ;
                          030B   706          ;   Node is a router. Test reachability of target.
                          030B   707          ;
           FCF2'  30   030B   708 95$:     BSBW    NET$TEST_REACH          ; Is node reachable ?
         0D 50    E9   030E   709          BLBC    R0,110$                 ; If LBC then no
                          0311   710          ;
                          0311   711          ;   If the remote node is an adjacent Phase II node, then
                          0311   712          ;   "tie" the logical link to the circuit for the life of
                          0311   713          ;   the logical link, thus making it "non-adaptive".
                          0311   714          ;
   02  51  10    10    ED   0311   715          CMPZV   #16,#16,R1,#ADJ$C_PTY_PH2 ; Is the remote a Phase II node?
             0F    12   0316   716          BNEQ    120$                    ; If NEQ no
          66   51    B0   0318   717          MOVW    R1,ICB$W_PATH(R6)       ; Else stuff the path ID
              007E   31   031B   718 100$:    BRW     140$                    ; Branch forward
                          031E   719
              00B7   31   031E   720 110$:    BRW     160$                    ; Take common exit
                          0321   721
          0FFC 8F    BA   0321   722 115$:    POPR    #^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ; Restore registers
               F4    11   0325   723          BRB     100$                    ; Branch "forward"
                          0327   724          ;
                          0327   725          ;   If the remote node is adjacent (hops=1), and the line buffer
                          0327   726          ;   size parameter is set higher than the executor buffer size,
                          0327   727          ;   then "tie" all logical links to the circuit for the life
                          0327   728          ;   of the logical link, thus making it "non-adaptive".  This
                          0327   729          ;   is so that the logical link can use a larger buffer size
                          0327   730          ;   for more optimal performance over the circuit.
                          0327   731          ;
FFFFFFFF 8F  51  10    10    EC   0327   732 120$:    CMPV    #16,#16,R1,#ADJ$C_PTY_UNK ; Is the node 1 hop away?
             E9    13   0330   733          BEQL    100$                    ; If not, skip it
          0FFC 8F    BB   0332   734          PUSHR   #^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ; Save registers
          58   51    3C   0336   735          MOVZWL  R1,R8                   ; Get ADJ index
             FCC4'  30   0339   736 125$:    BSBW    NET$FIND_ADJ            ; Lookup ADJ & LPD addresses
        59 50    E9   033C   737          BLBC    R0,130$                 ; Skip if not found for some reason
       55   56    D0   033F   738          MOVL    R6,R5                   ; Save LPD address
   58   28 A5    9A   0342   739          MOVZBL  LPD$B_PLVEC(R5),R8      ; Get PLVEC index
```

```
                5B   00000000'EF   D0  0346   740          MOVL    NET$GL_CNR_PLI,R1,      ; Point to line database
                           5A     D4  034D   741          CLRL    R10                     ; Starting at beginning
                                  034F   742          $SEARCH egl,pli,l,plvec         ; Search for corresponding line
                        37 50     E9  035E   743          BLBC    R0,130$                 ; Skip if none found
                                  0361   744          $GETFLD pli,l,bfs               ; Get line buffer size, if any
                        27 50     E9  036E   745          BLBC    R0,130$                 ; Skip if not set
                           58     25  C2  0371   746          SUBL    #TR$C_MAXHDR+NSP$C_MAXHDR,R8 ; Compute possible maximum
                                  0374   747                                          ; segment size
                51   00000000'EF   D0  0374   748          MOVL    NET$GL_PTR_VCB,R1       ; get address of RCB
                      7C A1    58   B1  037B   749          CMPW    R8,RCB$W_ECLSEGSIZ(R1)  ; check for segment size (R8) same
                                  037F   750                                          ; ... as executor buffer size
                           17     13  037F   751          BEQL    130$                    ; if equal, don't force fixed path
                                  0381   752          ;
                                  0381   753          ;  If an end node, DON'T lock the path (don't want to use DR).
                                  0381   754          ;
                    05    008A C1    91  0381   755          CMPB    RCB$B_ETY(R1),#ADJ$C_PTY_PH4N ; Is this node an end node?
                              10     13  0386   756          BEQL    130$                    ; If EQL, yes - don't force fixed path
                      56      10 AE    D0  0388   757          MOVL    4*4(SP),R6              ; Restore ICB address
                      66      20 A5    B0  038C   758          MOVW    LPD$W_PTH(R5),ICB$W_PATH(R6)    ; Stuff the path ID
                      12 A6      58    B0  0390   759          MOVW    R8,ICB$W_SEGSIZ(R6)     ; Set larger segment buffer size
                      06 A6      1E    B0  0394   760          MOVW    #30,ICB$W_TIM_INACT(R6) ; Lower inactivity timer (&& need symbol)
                         0FFC 8F      BA  0398   761 130$:    POPR    #^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ; Restore registers
                                  039C   762          ;
                                  039C   763          ;  If the node entry specifies an explicit output circuit, then
                                  039C   764          ;  force all I/O to use that circuit, overriding automatic routing.
                                  039C   765          ;
                00000004'EF       D5  039C   766 140$:    TSTL    NDI_PTR                 ; Is there an NDI block ?
                              31     13  03A2   767          BEQL    150$                    ; If EQL no, we're done
                                  03A4   768          $GETFLD ndi,s,nli               ; Get name of node's designated line
                           21 50     E9  03B1   769          BLBC    R0,150$                 ; If none specified use path 0
                5B   00000000'EF   D0  03B4   770          MOVL    NET$GL_CNR_CRI,R11      ; Get root of DLI list
                              5A     D4  03BB   771          CLRL    R10                     ; Indicate no current CNF
                                  03BD   772          $SEARCH egl,cri,s,nam           ; Find the CRI block
                              0A 50     E9  03CC   773          BLBC    R0,170$                 ; If LBC then not found
                            66    12 AA    B0  03CF   774          MOVW    CNF$W_ID(R10),ICB$W_PATH(R6) ; Establish the LPD i.d.
                              04     13  03D3   775          BEQL    170$                    ; If no LPD for this circuit, error
                                 50    00'  D0  03D5   776 150$:    MOVL    S^#SS$_NORMAL,R0        ; Indicate success
                                  03D8   777 160$:    RSB
                                  03D9   778
                           50    0000'8F      3C  03D9   779 170$:    MOVZWL  #SS$_DEVOFFLINE,R0      ; Loop circuit cannot be found
                              F8     11  03DE   780          BRB     160$
```

```
                        03E0   782 .SBTTL  PRS_ACCESS      - Parse NCB access control fields
                        03E0   783 ;+
                        03E0   784 ;
                        03E0   785 ; Parse the optional access control fields including the begining and
                        03E0   786 ; ending delimiter ('' only)
                        03E0   787 ;
                        03E0   788 ; INPUTS:        R6      ICB pointer
                        03E0   789 ;                R5      Pointer to 1st byte past NCB
                        03E0   790 ;                R4      Pointer to next byte to be parsed
                        03E0   791 ;
                        03E0   792 ;                All other regs are scratch
                        03E0   793 ;
                        03E0   794 ; OUTPUTS:       R6,R5   Preserved
                        03E0   795 ;                R4      Updated by number of bytes parsed
                        03E0   796 ;                R0      Routine status code
                        03E0   797 ;
                        03E0   798 ;                All other regs are garbage
                        03E0   799 ;
                        03E0   800 ;                ICB$B_ACCESS,ICB$T_ACCESS are setup if the optional
                        03E0   801 ;                fields are present
                        03E0   802 ;-
                        03E0   803 PRS_ACCESS:                                  ; Parse NCB access control fields
     64    22    91     03E0   804         CMPB    #^A'',(R4)                  ; Access control specified ?
           4A    12     03E3   805         BNEQ    20$                         ; If not, branch
           00    90     03E5   806         MOVB    #NMA$C_ACES_NONE,-          ; Disable proxy access
 0000001E'EF            03E7   807                 INT_B_PRX                   ;
           84    95     03EC   808         TSTB    (R4)+                       ; Skip over delimiter ('')
  58    3E A6    9E     03EE   809         MOVAB   ICB$T_ACCESS+1(R6),R8       ; Setup destination field - leave
                        03F2   810                                             ; room for count of first subfield
     59    3F    D0     03F2   811         MOVL    #ICB$C_ACCESS-1,R9          ; Setup size of dest field
                        03F5   812                                             ; C_ACCESS includes B_ACCESS
  53    00000230'EF    9E     03F5   813         MOVAB   NET$AB_ACC_TAB,R3       ; Setup translation table
                        03FC   814         ;
                        03FC   815         ;   Note that here ICB$B_ACCESS is cleared -- there was a -1 in it to
                        03FC   816         ;   signal "no access control yet".  If the user explicitly specifies
                        03FC   817         ;   null access control, e.g., node'"'::taskspecifier, then ICB$B_ACCESS
                        03FC   818         ;   will remain zero.  A -1 at the end of the parse would signal a need
                        03FC   819         ;   to supply the default access control.  It is important that null
                        03FC   820         ;   access control strings can be explicitly requested by the user
                        03FC   821         ;   so that the node receiving the connect can supply default inbound
                        03FC   822         ;   access info.
                        03FC   823         ;
        3C A6    94     03FC   824         CLRB    ICB$B_ACCESS(R6)            ; Init access string size
     5B    03    9A     03FF   825         MOVZBL  #3,R11                      ; Setup loop counter
           039D    30     0402   826 10$:     BSBW    GET_TOKEN               ; Get user id
     FF A8    57    90     0405   827         MOVB    R7,-1(R8)               ; Enter count of subfield
              57    96     0409   828         INCB    R7                      ; Account for count field
     3C A6    57    80     040B   829         ADDB    R7,ICB$B_ACCESS(R6)     ; Bump total bytes in strings
        58    57    C0     040F   830         ADDL    R7,R8                   ; Advance output pointer - note that
                        0412   831                                             ; R7 pts to first block after count
                        0412   832                                             ; for next subfield
     59    57    C2     0412   833         SUBL    R7,R9                       ; Adjust bytes left in buffer
  50    0000'8F    3C     0415   834         MOVZWL  #SS$_INVLOGIN,R0            ; Assume access fields too long
     57    27    B1     041A   835         CMPW    S^#NET$C_MAXACCFLD,R7       ; Access subfield within range?
              13    1F     041D   836         BLSSU   30$                         ; If GTRU then too large
        E0 5B    F5     041F   837         SOBGTR  R11,10$                     ; Get next string
           039E    30     0422   838         BSBW    SCAN_BLANKS                 ; Scan blanks and tabs
```

```
                50  0000'8F   3C  0425   839          MOVZWL  #SS$_IVDEVNAM,R0       ; Assume NCB format error
                    84   22   91  042A   840          CMPB    #^A'"',(R4)+           ; Is next character a quote ?
                         03   12  042D   841          BNEQ    30$                    ; Illegal NCB if NEQ
                    50   00'  D0  042F   842 20$:     MOVL    S^#SS$_NORMAL,R0       ; Indicate success
                              05  0432   843 30$:     RSB
```

```
                          0433    845 .SBTTL  PRS_OBJECT      - Parse NCB target task identifier
                          0433    846 ;+
                          0433    847 ;
                          0433    848 ; The taskname specifier is parsed, the OBI block located, and the
                          0433    849 ; ICB destination task fields setup.  The legal taskname formats are:
                          0433    850 ;
                          0433    851 ;       ''objectname=
                          0433    852 ;       ''objectnumber=
                          0433    853 ;       ''TASK=taskname
                          0433    854 ;       ''0=taskname
                          0433    855 ;
                          0433    856 ; The parse includes the parse of the leading '' but does not include
                          0433    857 ; the terminating delimiter since it may vary.
                          0433    858 ;
                          0433    859 ; INPUTS:       R6      ICB pointer
                          0433    860 ;               R5      Points past NCB
                          0433    861 ;               R4      Points to next unparsed byte in NCB
                          0433    862 ;
                          0433    863 ;               All other registers are scratch
                          0433    864 ;
                          0433    865 ; OUTPUTS:      R6,R5   Preserved
                          0433    866 ;               R4      Updated to point to next unparsed byte
                          0433    867 ;               R0      Routine status
                          0433    868 ;               All other registers are garbage
                          0433    869 ;
                          0433    870 ;               ICB destination task fields are setup
                          0433    871 ;
                          0433    872 ;               OBI_PTR points the OBI CNF
                          0433    873 ;                       0 if taskname specified by number and the
                          0433    874 ;                         corresponding OBI entry is not found
                          0433    875 ;-
                          0433    876 PRS_OBJECT:                                 ; Parse NCB target taskname
          0000000C'EF     7C    0433    877         CLRQ    OBJ_Q_DESC            ; Init the object descriptor
          00000014'EF     7C    0439    878         CLRQ    TSK_Q_DESC            ; Init the task descriptor
     5B   00000000'EF     D0    043F    879         MOVL    NET$GL_CNR_OBI,R11    ; Setup root of OBI list
                          0446    880         ;
                          0446    881         ;
                          0446    882         ;   Locate begining of object specifier
                          0446    883         ;
                    037A  30    0446    884         BSBW    SCAN_BLANKS           ; Skip blanks and tabs
                 84   22  91    0449    885         CMPB    #^A'"',(R4)+          ; Correct delimiter
                      6E  12    044C    886         BNEQ    17$                   ; If NEQ no, may be some other field
                          044E    887         ;
                          044E    888         ;
                          044E    889         ;   Locate object name or number -- that part before the "=" delimiter
                          044E    890         ;
                   59 0C  9A    044E    891         MOVZBL  S^#NET$C_MAXOBJNAM,R9 ; Set max field size
          58   00000000'EF     D0    0451    892         MOVL    NET$GL_UTLBUF,R8      ; Setup output buffer address
          00000010'EF   54     D0    0458    893         MOVL    R4,OBJ_Q_DESC+4       ; Point to begining of object specifier
                    0310  30    045F    894         BSBW    GET_STR_NUM           ; Get ascii string or binary value
0000000C'EF  54   00000010'EF     C3    0462    895         SUBL3   OBJ_Q_DESC+4,R4,- ; Complete descriptor by calculating
                          046E    896                         OBJ_Q_DESC            ; the string size
                      5A  D4    046E    897         CLRL    R10                   ; Indicate no current CNF
                      50  D4    0470    898         CLRL    R0                    ; Preset return error flag
                 22   51  E9    0472    899         BLBC    R1,10$                ; Br unless specified by number
          000000FF 8F   52  D1    0475    900         CMPL    R2,#NET$C_MAX_OBJ     ; Object # within range ?
                      3B  1A    047C    901         BGTRU   15$                   ; If GTRU then out of range
```

```
                                047E   902          ;
                                047E   903          ;
                                047E   904          ;   Locate OBI block.  This block is not required if the object number
                                047E   905          ;   was specified and it was non-zero.  Else it is needed to continue.
                                047E   906          ;
                     58   52    D0   047E   907          MOVL    R2,R8                   ; Setup search key value
                                       0481   908          $SEARCH egl,obi,l,num           ; Find the matching OBI block
                       2C 50    E8   0490   909          BLBS    R0,20$                  ; If LBS then it was found
                          5A    D4   0493   910          CLRL    R10                     ; Else nullify OBI CNF pointer
                          28    11   0495   911          BRB     20$                     ; Continue in common
                                       0497   912 10$:     $SEARCH egl,obi,s,nam           ; Find the macthing OBI CNF
                       10 50    E9   04A6   913          BLBC    R0,15$                  ; If LBC then not found
                                       04A9   914          $GETFLD obi,l,num               ; Get the number
                       06 50    E8   04B6   915          BLBS    R0,20$                  ; Okay if LBS
                        00CB    31   04B9   916 15$:     BRW     200$                    ; Else, exit with "no such object"
                        00CE    31   04BC   917 17$:     BRW     300$                    ; Exit with "invalid device (NCB) name"
          00000008'EF    5A    D0   04BF   918 20$:     MOVL    R10,OBI_PTR             ; Setup CNF pointer
                                04C6   919          ;
                                04C6   920          ;
                                04C6   921          ;   Make sure an "=" sign follows the object specifier
                                04C6   922          ;
                          02FA  30   04C6   923          BSBW    SCAN_BLANKS             ; Skip over blanks and tabs
                     84    3D   91   04C9   924          CMPB    #^A'=',(R4)+            ; Is correct delimiter there ?
                          EE   12   04CC   925          BNEQ    17$                     ; If NEQ then incorrect
                                04CE   926          ;
                                04CE   927          ;
                                04CE   928          ;   Setup the ICB remote task desciption
                                04CE   929          ;
                       29 A6    94   04CE   930          CLRB    ICB$B_DSTFMT(R6)        ; Assume format type zero
                       2B A6    94   04D1   931          CLRB    ICB$T_DSTDSC(R6)        ; Nullify ascii object string
                 28 A6    02   90   04D4   932          MOVB    #2,ICB$B_RPRNAM(R6)     ; Account for format,object type
                 2A A6    58   90   04D8   933          MOVB    R8,ICB$B_DSTOBJ(R6)     ; Enter object type
                          4A   12   04DC   934          BNEQ    40$                     ; If NEQ then type is not TASK
                 29 A6    01   90   04DE   935          MOVB    #1,ICB$B_DSTFMT(R6)     ; Format type 1
          53   00000130'EF    9E   04E2   936          MOVAB   NET$AB_OBJTRAN,R3       ; Setup translation table
                 58   2C A6    9E   04E9   937          MOVAB   ICB$T_DSTDSC+1(R6),R8   ; Setup dest. string pointer
                    59   10    D0   04ED   938          MOVL    #ICB$C_RPRNAM-4,R9      ; Setup size of dest. field
                                04F0   939                                           ; (-3 for DSTFMT,DSTOBJ, taskname
                                04F0   940                                           ;  count and ICB$_RPRNAM fields)
                        02AF    30   04F0   941          BSBW    GET_TOKEN               ; Scan blanks and move string
          00000014'EF    57    7D   04F3   942          MOVQ    R7,TSK_Q_DESC           ; Setup taskname descriptor
                 2B A6    57   90   04FA   943          MOVB    R7,ICB$T_DSTDSC(R6)     ; Store taskname length in ICB
                          03   12   04FE   944          BNEQ    30$                     ; If not null then good task i.d.
                        0084   31   0500   945          BRW     200$                    ; Else, illegal task i.d.
           28 A6    57   03   81   0503   946 30$:     ADDB3   #3,R7,ICB$B_RPRNAM(R6)  ; Set total RPRNAM length
                                0508   947          ;
                                0508   948          ;
                                0508   949          ;   The connect is to object number 0.
                                0508   950          ;
                                0508   951          ;   Since there may be many OBI entries for object number 0 (TASK),
                                0508   952          ;   see if there is one which matches the qualifying taskname.  If so,
                                0508   953          ;   use it instead of the generic TASK OBI.
                                0508   954          ;
                             5A   DD   0508   955          PUSHL   R10                     ; Save the TASK OBI
                             5A   D4   050A   956          CLRL    R10                     ; Nullify OBI CNF pointer
                                       050C   957          $SEARCH egl,obi,s,nam           ; See if there's an OBI with this name
                       03 50    E9   051B   958          BLBC    R0,35$                  ; If LBC then no
```

```
                6E   5A   D0  051E   959              MOVL    R10,(SP)                ; Overly the OBI pointer on the stack
     00000008'EF 8ED0   0521   960 35$:    POPL    OBI_PTR                 ; Update the official OBI pointer
                        0528   961 40$:    ;
                        0528   962          ;
                        0528   963          ;   Setup the proxy login state for this OBI
                        0528   964          ;
5A   00000008'EF   D0   0528   965          MOVL    OBI_PTR,R10             ; Get the OBI
                17   13   052F   966          BEQL    60$                     ; If EQL then none
                        0531   967          $GETFLD obi,l,prx               ; Get proxy login state
              07 50   E9   053E   968          BLBC    R0,60$                  ; If LBC then none
0000001D'EF   58   90   0541   969          MOVB    R8,OBI_B_PRX            ; Else override the default
                        0548   970 60$:    ;
                        0548   971          ;
                        0548   972          ;   Setup the remote user i.d. (RID) for display purposes.  If the
                        0548   973          ;   target number is zero then use the taskname from the NCB.  Else,
                        0548   974          ;   use the object name from the OBI -- if no OBI use the object
                        0548   975          ;   name/number from the NCB.
                        0548   976          ;
                3E   BB   0548   977          PUSHR   #^M<R1,R2,R3,R4,R5>     ; Save regs
57   00000014'EF   7D   054A   978          MOVQ    TSK_Q_DESC,R7           ; Setup taskname descriptor assuming
                        0551   979                                          ; object type 0
 50    2A A6   90   0551   980          MOVB    ICB$B_DSTOBJ(R6),R0     ; Get object number
                1D   13   0555   981          BEQL    80$                     ; If EQL then use taskname
57   0000000C'EF   7D   0557   982          MOVQ    OBJ_Q_DESC,R7           ; Get object name/number descriptor
5A   00000008'EF   D0   055E   983          MOVL    OBI_PTR,R10             ; Get OBI pointer
                0D   13   0565   984          BEQL    80$                     ; If EQL none, use object name/number
                        0567   985                                          ; from NCB
                        0567   986          $GETFLD obi,s,nam               ; Else use object name from NCB
     0092 C6   57   90   0574   987 80$:    MOVB    R7,ICB$B_RID(R6)        ; Setup text field length
     20   68   57   2C   0579   988          MOVC5   R7,(R8),#^A" ",-        ; Move the text
     0093 C6   10        057D   989                  #ICB$C_RID,ICB$T_RID(R6);
                3E   BA   0581   990          POPR    #^M<R1,R2,R3,R4,R5>     ; Restore regs
                        0583   991          ;
                        0583   992          ;
                        0583   993          ;   Done, return to caller
                        0583   994          ;
     50   00'   D0   0583   995          MOVL    S^#SS$_NORMAL,R0        ; Indicate success
                05   0586   996 100$:    RSB                             ; Done
                        0587   997          ;
50   0000'8F   3C   0587   998 200$:    MOVZWL  #SS$_NOSUCHOBJ,R0       ; Indicate error
                05   058C   999          RSB                             ; Done
                        058D  1000          ;
50   0000'8F   3C   058D  1001 300$:    MOVZWL  #SS$_IVDEVNAM,R0        ; Assume NCB format error
                05   0592  1002          RSB                             ; Done
```

```
                                0593  1004 .SBTTL  PRS_END - Parse the remainder of the NCB
                                0593  1005 ;+
                                0593  1006 ;
                                0593  1007 ; Find the link i.d. and optional data.  If none specified then this is
                                0593  1008 ; a "connect initiate".
                                0593  1009 ;
                                0593  1010 ; *** tbs ****  (R4 -> next input char,  R5 -> past end of NCB)
                                0593  1011 ;-
                                0593  1012 PRS_END:                                      ; Parse remainder of the NCB
                  7C A6   94    0593  1013          CLRB    ICB$B_DATA(R6)              ; Assume no optional data
                  02 A6   B4    0596  1014          CLRW    ICB$W_LOCLNK(R6)            ; Assume connect initiate
                    0227  30    0599  1015          BSBW    SCAN_BLANKS                 ; Scan past tabs,blanks
                64    2F  91    059C  1016          CMPB    #^A"7",(R4)                 ; Is the 'tail' of the NCB here
                      08  13    059F  1017          BEQL    5$                          ; If EQL yes, parse it
                64    22  91    05A1  1018          CMPB    #^A"""",(R4)                ; Is NCB delimiter next?
                      29  13    05A4  1019          BEQL    10$                         ; If EQL yes, check for end of NCB
                  0034  31    05A6  1020          BRW     20$                         ; Else NCB is malformed
                      84  95    05A9  1021 5$:      TSTB    (R4)+                       ; Skip over "/"
             02 A6    84  B0    05AB  1022          MOVW    (R4)+,ICB$W_LOCLNK(R6)      ; Enter local link id
                64    22  91    05AF  1023          CMPB    #^A"""",(R4)                ; Is NCB delimiter next ?
                      1B  13    05B2  1024          BEQL    10$                         ; If EQL yes, chk for legal NCB
          50  0000'8F   3C    05B4  1025          MOVZWL  #SS$_TOOMUCHDATA,R0         ; Assume error
                51    64  9A    05B9  1026          MOVZBL  (R4),R1                     ; Get optional data count field
                10    51  91    05BC  1027          CMPB    R1,#16                      ; Check length of optional data
                      1C  1A    05BF  1028          BGTRU   20$                         ; Br if too long
                      51  D6    05C1  1029          INCL    R1                          ; Include the count field
                      30  BB    05C3  1030          PUSHR   #^M<R4,R5>                  ; Save critical regs
       7C A6    64    51  28    05C5  1031          MOVC    R1,(R4),ICB$B_DATA(R6)      ; Move optional data
                54    51  D0    05CA  1032          MOVL    R1,R4                       ; Get next character in NCB
                      30  BA    05CD  1033          POPR    #^M<R4,R5>                  ; Restore regs
                                05CF  1034          ;
                                05CF  1035          ;   Check to see if the NCB is terminated correctly.  This means that
                                05CF  1036          ;   we must be at the last character in the NCB and it must be a double
                                05CF  1037          ;   quote.  However, if the user is doing a "transparent" $ASSIGN to
                                05CF  1038          ;   SYS$NET, then there is some garbage containing local the task
                                05CF  1039          ;   specification after the optional data -- ignore it.
                                05CF  1040          ;
                                05CF  1041          ;   The actual test used to verify a correct NCB is to check that there
                                05CF  1042          ;   is a """ character somewhere between the current pointer and the
                                05CF  1043          ;   end of the NCB.  This is simple and more forgiving of user error.
                                05CF  1044          ;
          55    54    D1    05CF  1045 10$:     CMPL    R4,R5                       ; Are we beyond the end ?
                09    1E    05D2  1046          BGEQU   20$                         ; If so, NCB format error
                84    22    91    05D4  1047          CMPB    #^A"""",(R4)+               ; Is NCB delimiter there ?
                      F6    12    05D7  1048          BNEQ    10$                         ; If not, continue search
                50    00'   D0    05D9  1049          MOVL    S^#SS$_NORMAL,R0            ; Indicate success
                            05    05DC  1050          RSB
                                05DD  1051
          50  0000'8F   3C    05DD  1052 20$:     MOVZWL  #SS$_IVDEVNAM,R0            ; Signal illegal NCB
                            05    05E2  1053          RSB
```

```
                        05E3  1055 CHECK_ACCESS:                          ; See if access is allowed to node
53  00000000'EF   D0    05E3  1056          MOVL    NET$GL_SAVE_IRP,R3     ; Get the IRP address
          12   E0    05EA  1057          BBS     #PRV$V_OPER,-          ; If user has OPER then the connect is
        3E 40 A3        05EC  1058                  IRP$Q_NT_PRVMSK(R3),100$; always allowed -- bypass all checks
                        05EF  1059          ;
                        05EF  1060          ;
                        05EF  1061          ;   Check to see if the connect is allowed based on the state of the
                        05EF  1062          ;   local node.
                        05EF  1063          ;
                        05EF  1064          ;       state       Allow connect if
                        05EF  1065          ;       --------    ----------------
                        05EF  1066          ;       ON          always
                        05EF  1067          ;       RESTRICT    if this is a connect initiate, or
                        05EF  1068          ;                   if the partner node is the local node
                        05EF  1069          ;       SHUT        never
                        05EF  1070          ;       OFF         never
                        05EF  1071          ;
                        05EF  1072          ;
50  00000000'EF   D0    05EF  1073          MOVL    NET$GL_PTR_VCB,R0      ; Get the RCB address
      50   61 A0   9A    05F6  1074          MOVZBL  RCB$B_STI(R0),R0       ; Get the local node state
         50   01   91    05FA  1075          CMPB    S^#ACP$C_STA_N,R0      ; Is state "ON"?
              10   13    05FD  1076          BEQL    10$                    ; If EQL yes - no local restrictions
         50   02   91    05FF  1077          CMPB    S^#ACP$C_STA_R,R0      ; Is state "RESTRICTED"?
              2D   12    0602  1078          BNEQ    200$                   ; If NEQ no, connect not allowed
       008D C6    B5    0604  1079          TSTW    ICB$W_REMNOD(R6)       ; Is it for the local node?
              23   13    0608  1080          BEQL    100$                   ; If EQL yes - connect OK
        02 A6     B5    060A  1081          TSTW    ICB$W_LOCLNK(R6)       ; Connect initiate?
              22   12    060D  1082          BNEQ    200$                   ; If NEQ no - connect not allowed
                        060F  1083 10$:     ;
                        060F  1084          ;
                        060F  1085          ;   Check to see if the connect is allowed based on the local access
                        060F  1086          ;   restrictions set for the remote node.
                        060F  1087          ;
                        060F  1088          $DISPATCH  TYPE=B,NDI_B_ACC -  ;
                        060F  1089          <-
                        060F  1090              <NMA$C_ACES_NONE, 200$> -  ; No access allowed
                        060F  1091              <NMA$C_ACES_INCO,  60$> -  ; Inbound access allowed
                        060F  1092              <NMA$C_ACES_OUTG,  50$> -  ; Outbound access allowed
                        060F  1093              <NMA$C_ACES_BOTH, 100$> -  ; All access allowed
                        060F  1094          >
            0C    11    061F  1095          BRB     100$                   ; Code is not recognized, ignore it
        02 A6     B5    0621  1096 50$:     TSTW    ICB$W_LOCLNK(R6)       ; No inbound access. Connect confirm ?
              0B   12    0624  1097          BNEQ    200$                   ; If NEQ then yes, access not allowed
              05   11    0626  1098          BRB     100$                   ; Else report success
        02 A6     B5    0628  1099 60$:     TSTW    ICB$W_LOCLNK(R6)       ; No outbound access. Connect initiate?
              04   13    062B  1100          BEQL    200$                   ; If EQL then yes, access not allowed
      50   00'    D0    062D  1101 100$:    MOVL    S^#SS$_NORMAL,R0       ; Indicate success
                  05    0630  1102          RSB
                        0631  1103
                        0631  1104 200$:    ;
                        0631  1105          ;   The connection is not allowed.  Tell NETDRIVER to terminate the
                        0631  1106          ;   link.  Return an error message to our caller.
                        0631  1107          ;
    53   02 A6     3C    0631  1108          MOVZWL  ICB$W_LOCLNK(R6),R3    ; Setup local link number
        52   03    3C    0635  1109          MOVZWL  #NET$C_DR_SHUT,R2      ; Setup disconnect reason
51  00000000'EF   D0    0638  1110          MOVL    NET$GL_SAVE_IRP,R1     ; Get user's IRP
    51   0C A1    D0    063F  1111          MOVL    IRP$L_PID(R1),R1       ; Setup user's PID
```

```
                00000000'EF   16  0643  1112          JSB     NET$CONNECT_FAIL        ; Report connect failure to NETDRIVER
           50      0000'8F    3C  0649  1113          MOVZWL  #SS$_SHUT,R0            ; Signal connects not allowed
                              05  064E  1114          RSB
```

```
                                064F  1116 .SBTTL  DFLT_ACCESS - Get default access control
                                064F  1117 ;+
                                064F  1118 ;
                                064F  1119 ; Use the default information from the NDI block.
                                064F  1120 ;
                                064F  1121 ;-
                                064F  1122 DFLT_ACCESS:                                 ; Get default access control
                     3C A6   94 064F  1123          CLRB    ICB$B_ACCESS(R6)         ; Init access string length
00000000'EF   00000010'EF   9E 0652  1124          MOVAB   NONPRV_TAB,ACC_TAB       ; Assume no privileges needed
      5A      00000008'EF   D0 065D  1125          MOVL    OBI_PTR,R10              ; Get OBI CNF pointer
                        03  12 0664  1126          BNEQ    10$                      ; If NEQ then OBI exists
                      0101  31 0666  1127          BRW     100$                     ;
                            0669  1128 10$:       $GETFLD obi,l,lpr                ; Get tne high order priv mask field
         00000000'FF    58   D0 0676  1129          MOVL    R8,@NET$GL_UTLBUF        ; Save the low order priority mask
                            067D  1130          $GETFLD obi,l,hpr                ; Get the low order priv mask field
      5A   00000000'EF   D0 068A  1131          MOVL    NET$GL_UTLBUF,R10        ; Point to the utility buffer
            04 AA    58   D0 0691  1132          MOVL    R8,4(R10)                ; Save the high order priority mask
                            0695  1133
                            0695  1134          ASSUME  PRV$V_NETMBX    LT 32
                            0695  1135          ASSUME  PRV$V_TMPMBX    LT 32
                            0695  1136
            00 6A    0F   E5 0695  1137          BBCC    #PRV$V_TMPMBX,(R10),20$  ; Zero non-priv bits
            00 6A    14   E5 0699  1138 20$:     BBCC    #PRV$V_NETMBX,(R10),30$  ;
               50    6A   7D 069D  1139 30$:     MOVQ    (R10),R0                 ; Get required privilege mask
                     23   13 06A0  1140          BEQL    40$                      ; If EQL then none needed
      53   00000000'EF   D0 06A2  1141          MOVL    NET$GL_SAVE_IRP,R3       ; Get current IRP pointer
               50   40 A3   CA 06A9  1142          BICL    IRP$Q_NT_PRVMSK(R3),R0   ; Test for required privileges
                     13   12 06AD  1143          BNEQ    35$                      ; Br if user lacks privilege
               51   44 A3   CA 06AF  1144          BICL    IRP$Q_NT_PRVMSK+4(R3),R1 ; Test high order part of mask
                     0D   12 06B3  1145          BNEQ    35$                      ; Br if user lacks privilege
00000000'EF   00000000'EF   9E 06B5  1146          MOVAB   PRV_TAB,ACC_TAB          ; Setup for priv access
                     03   11 06C0  1147          BRB     40$                      ; Continue
                   00A5   31 06C2  1148 35$:     BRW     100$                     ; No default access control
                          06C5  1149          ;
                          06C5  1150          ;
                          06C5  1151          ;  Get NDI to use for default access control.  If no NDI is
                          06C5  1152          ;  currently specified then there's no default.
                          06C5  1153          ;
      5B   00000000'EF    D0 06C5  1154 40$:     MOVL    NET$GL_CNR_NDI,R11       ; Get NDI root pointer
      5A   00000004'EF    D0 06CC  1155          MOVL    NDI_PTR,R10              ; Get NDI CNF pointer
                     ED   13 06D3  1156          BEQL    35$                      ; Br if no NDI block
                          06D5  1157          ;
                          06D5  1158          ;  If the NDI is a
                          06D5  1159          ;  loopnode NDI and its access control is null, use the access control
                          06D5  1160          ;  of the NDI with the matching address and which is not a loopnode
                          06D5  1161          ;  (currently this can only be the local NDI).  If there is no such
                          06D5  1162          ;  NDI then there is no default access control.
                          06D5  1163          ;
                          06D5  1164          $GETFLD ndi,v,loo                ; Loopnode ?
                  39 58   E9 06E2  1165          BLBC    R8,60$                   ; If loopnode,
      59   00000000'FF    D0 06E5  1166          MOVL    @ACC_TAB,R9              ; Setup first field (user) id
           00000000'EF    16 06EC  1167          JSB     CNF$GET_FIELD            ; Get the USER_ID field
                  29 50   E8 06F2  1168          BLBS    R0,60$                   ; If LBS then non-null, use it
         58   008D C6    3C 06F5  1169 50$:     MOVZWL  ICB$W_REMNOD(R6),R8      ; Get node address
                     5A   D4 06FA  1170          CLRL    R10                      ; Indicate no current CNF
                          06FC  1171          $SEARCH  eql,ndi,l,add           ; Find CNF with matching address
                  5C 50   E9 070B  1172          BLBC    R0,100$                  ; No default access if no NDI
```

```
                        070E  1173          $GETFLD ndi,v,loo                       ; Loopnode ?
                4C 58   E8  071B  1174          BLBS    R8,100$                 ; If LBS its a loopnode - can't use it
                        071E  1175                                                  ; Loop nodes are stored in the list
                        071E  1176                                                  ; last and so there's no use searching
                        071E  1177                                                  ; any further
                        071E  1178 60$:         ;
                        071E  1179              ;   If this connect is for the local node, and we have determined
                        071E  1180              ;   that the non-privileged account is to be used, then don't provide
                        071E  1181              ;   any default outbound access control, but instead, rely on the
                        071E  1182              ;   access control being defaulted on the incoming side.  This is
                        071E  1183              ;   to avoid conflict with the proxy mechanism for executor connects.
                        071E  1184              ;
                12 AA   B5  071E  1185          TSTW    CNF$W_ID(R10)           ; Is this the local node?
                   10   12  0721  1186          BNEQ    70$                     ; Skip if not
    50  00000010'EF   9E  0723  1187          MOVAB   NONPRV_TAB,R0           ; Get address of non-priv param table
    50  00000000'EF   D1  072A  1188          CMPL    ACC_TAB,R0              ; Is connect non-priv or privileged?
                   37   13  0731  1189          BEQL    100$                    ; If local non-priv connect, no default
                        0733  1190 70$:         ;
                        0733  1191              ;
                        0733  1192              ;   Move access control strings
                        0733  1193              ;
                   30   BB  0733  1194          PUSHR   #^M<R4,R5>              ; Save critical regs
            53  3D A6   9E  0735  1195          MOVAB   ICB$T_ACCESS(R6),R3     ; Get output pointer
    59  00000000'FF   D0  0739  1196 80$:       MOVL    @ACC_TAB,R9             ; Get field i.d.
                   26   13  0740  1197          BEQL    90$                     ; Done if EQL
00000000'EF        04   C0  0742  1198          ADDL    #4,ACC_TAB              ; Bump the pointer
     00000000'EF   16  0749  1199          JSB     CNF$GET_FIELD           ; Get the string descriptor
          3C A6    57  074F  1200          ADDB    R7,ICB$B_ACCESS(R6)     ; Update total size
             3C A6 96  0753  1201          INCB    ICB$B_ACCESS(R6)        ; Account for count byte
             40 8F 91  0756  1202          CMPB    #ICB$C_ACCESS,-         ; Can it fit ?
             3C A6     0759  1203                  ICB$B_ACCESS(R6)        ;
                11   19  075B  1204          BLSS    200$                    ; If LSS no, must be bug
          83    57   90  075D  1205          MOVB    R7,(R3)+                ; Enter count field
                D7   13  0760  1206          BEQL    80$                     ; If EQL then get next string
    63    68    57   28  0762  1207          MOVC3   R7,(R8),(R3)            ; Enter string
                D1   11  0766  1208          BRB     80$                     ; Loop
                30   BA  0768  1209 90$:       POPR    #^M<R4,R5>              ; Restore regs
                     076A  1210              ;
          50    00'  D0  076A  1211 100$:      MOVL    S^#SS$_NORMAL,R0        ; Always successful
                05   076D  1212              RSB
                     076E  1213
                     076E  1214 200$:       BUG_CHECK NETNOSTATE,FATAL      ; Bugcheck
```

```
                     0772  1216 .SBTTL  GET_STR_NUM - Get next numeric token
                     0772  1217 ;+
                     0772  1218 ;
                     0772  1219 ; The next string is scanned until the first non-numeric, non-alphabetic
                     0772  1220 ; ascii character.  All lower case alphabetics are converted to upper
                     0772  1221 ; case.  Leading blanks and tabs are skipped.  If the string contains
                     0772  1222 ; all ascii numeric characters, it is converted from its ascii-decimal
                     0772  1223 ; form to binary.
                     0772  1224 ;
                     0772  1225 ; INPUTS:       R9  Maximum allowed output length
                     0772  1226 ;                R8  Pointer to input buffer
                     0772  1227 ;
                     0772  1228 ;                R7,R3-R0 Scratch
                     0772  1229 ;
                     0772  1230 ; OUTPUTS:      R7  Number of characters in output buffer
                     0772  1231 ;                R4  Pointer to next unparsed byte in input stream
                     0772  1232 ;                R3  Garbage
                     0772  1233 ;                R2  Converted ascii value if R1 has low bit set,
                     0772  1234 ;                    zero if R7=0
                     0772  1235 ;                R1  Low bit set if string was all numeric or null
                     0772  1236 ;                R0  Garbage
                     0772  1237 ;
                     0772  1238 ;                All other registers are preserved.
                     0772  1239 ;-
                     0772  1240 GET_STR_NUM:                                 ; Get string or number
53   00000030'EF  9E 0772  1241         MOVAB   NET$AB_UPASCNUM,R3          ; Setup translation table
             27  10 0779  1242         BSBB    GET_TOKEN                   ; Get the translated string
             52  D4 077B  1243         CLRL    R2                          ; Zero string converted value
       51    57  D0 077D  1244         MOVL    R7,R1                       ; Any characters in moved ?
             1A  13 0780  1245         BEQL    15$                         ; Br if none moved
       53    58  D0 0782  1246         MOVL    R8,R3                       ; Get ptr to first character
  50   83    30  83 0785  1247 10$:     SUBB3   #^A"0",(R3)+,R0             ; Get binary of character
             14  19 0789  1248         BLSS    20$                         ; Br if non-numeric
       09    50  91 078B  1249         CMPB    R0,#9                       ; Test upper bound
             0F  14 078E  1250         BGTR    20$                         ; Br if non-numeric
       50    50  9A 0790  1251         MOVZBL  R0,R0                       ; Zero garbage bytes
       52    0A  C4 0793  1252         MULL    #10,R2                      ; Multiply old value by ten
       52    50  C0 0796  1253         ADDL    R0,R2                       ; and add new increment
     E9      51  F5 0799  1254         SOBGTR  R1,10$                      ; Loop for each character
             51  D6 079C  1255 15$:     INCL    R1                          ; Flag 'all numeric string'
                 05 079E  1256         RSB
             51  D4 079F  1257 20$:     CLRL    R1                          ; Flag 'non-numeric'
                 05 07A1  1258         RSB
```

```
                                                     07A2  1260 .SBTTL  GET_TOKEN - Get next token
                                                     07A2  1261 ;+
                                                     07A2  1262 ;
                                                     07A2  1263 ; The input stream is scanned until a delimiter is found.  A delimiter
                                                     07A2  1264 ; is defined as any character which the translation table translates
                                                     07A2  1265 ; to a zero.  The input pointer is advanced up to, but not past, the
                                                     07A2  1266 ; delimiter.  All leading blanks and tabs are skipped over.
                                                     07A2  1267 ;
                                                     07A2  1268 ; INPUTS:  R9      Max size of input string
                                                     07A2  1269 ;          R8      Address of buffer to receive output
                                                     07A2  1270 ;          R7      Scratch
                                                     07A2  1271 ;          R6      ICB pointer
                                                     07A2  1272 ;          R5      Points past NCB
                                                     07A2  1273 ;          R4      Next character in input string
                                                     07A2  1274 ;          R3      Translation table address
                                                     07A2  1275 ;          R2-R0   Scratch
                                                     07A2  1276 ;
                                                     07A2  1277 ; OUTPUTS: R7      Number of characters in output buffer
                                                     07A2  1278 ;          R4      Points to first unmoved character
                                                     07A2  1279 ;          R2-R0   Garbage
                                                     07A2  1280 ;
                                                     07A2  1281 ;          All other registers are preserved.
                                                     07A2  1282 ;-
                                                     07A2  1283 GET_TOKEN:                                   ; Move input up to delimiter
                                           1F    10  07A2  1284          BSBB    SCAN_BLANKS               ; Skip blanks and tabs
                                           55    DD  07A4  1285          PUSHL   R5                        ; Protect regs form MOVTUC
                                     50    55    54  C3  07A6  1286          SUBL3   R4,R5,R0                  ; Get bytes left in input stream
68   59   63   00   64    50    2F  07AA  1287          MOVTUC  R0,(R4),#0,(R3),R9,(R8) ; Translate/move the string
                                     54    51    D0  07B1  1288          MOVL    R1,R4                     ; Get input stream pointer
                                     57    55    58  C3  07B4  1289          SUBL3   R8,R5,R7                  ; Get # of bytes moved
                                           55 8ED0  07B8  1290          POPL    R5                        ; Recover regs
                                              05  07BB  1291          RSB
                                                     07BC  1292 ;+
                                                     07BC  1293 ; SCAN_BLANKS - Skip over blank and tab characters
                                                     07BC  1294 ;
                                                     07BC  1295 ; The input stream is advanced to the first non blank/tab character.
                                                     07BC  1296 ;
                                                     07BC  1297 ; INPUTS:  R5   Points to first character beyond input stream
                                                     07BC  1298 ;          R4   Points to next character in input stream
                                                     07BC  1299 ; OUTPUTS: R4   Points to next non blank/tab character in input stream
                                                     07BC  1300 ;-
                                                     07BC  1301                      .ENABL  LSB
55   54   D1   07BC  1302 10$:     CMPL    R4,R5                     ; At the end of input stream ?
     10   1E   07BF  1303          BGEQU   20$                       ; If so, branch
     54   D6   07C1  1304          INCL    R4                        ; Advance input pointer
                                              07C3  1305
                                              07C3  1306 SCAN_BLANKS:                           ; Skip over blanks and tabs
     64   95   07C3  1307          TSTB    (R4)                      ; Is character null?
     F5   13   07C5  1308          BEQL    10$                       ; If so, skip it
64   20   91   07C7  1309          CMPB    #SPACE,(R4)               ; Is character a space ?
     F0   13   07CA  1310          BEQL    10$                       ; If so then loop
64   09   91   07CC  1311          CMPB    #TAB,(R4)                 ; Is it a tab ?
     EB   13   07CF  1312          BEQL    10$                       ; If so then loop
              05   07D1  1313 20$:     RSB
                                              07D2  1314                      .DSABL  LSB
                                              07D2  1315
                                              07D2  1316 .END
```

```
$$T1                    = 00000001
$$_NSPMSG               = 00000000
$$_TR3MSG               = 00000000
$$_TR4MSG               = 00000000
ABD$C_LENGTH            = 00000008
ABD$C_NAME              = 00000002
ABD$W_COUNT             = 00000002
ABD$W_TEXT              = 00000000
ACCESS_DONE               00000157 R      04
ACC_TAB                   00000000 R      03
ACP$C_STA_F             = 00000004
ACP$C_STA_H             = 00000005
ACP$C_STA_I             = 00000000
ACP$C_STA_N             = 00000001
ACP$C_STA_R             = 00000002
ACP$C_STA_S             = 00000003
ADJ$C_PTY_PH2           = 00000002
ADJ$C_PTY_PH4N          = 00000005
ADJ$C_PTY_UNK           = FFFFFFFF
BIN_HEXASC                00000020 R      02
BIT...                  = 00000006
BUG$_NETNOSTATE           ********    X   04
C                       = 00000001
CALL_NETDRIVER            ********    X   04
CHECK_ACCESS              000005E3 R      04
CNF$GET_FIELD             ********    X   04
CNF$KEY_SEARCH            ********    X   04
CNF$W_ID                = 00000012
CNF$_ADVANCE            = 00000000
CNF$_QUIT               = 00000002
CNF$_TAKE_CURR          = 00000003
CNF$_TAKE_PREV          = 00000001
CNR$C_FLINK             = 00000000
DFLT_ACCESS               0000064F R      04
EXE$IPID_TO_EPID          ********    X   04
GET_STR_NUM               00000772 R      04
GET_TOKEN                 000007A2 R      04
ICB$B_ACCESS            = 0000003C
ICB$B_DATA              = 0000007C
ICB$B_DSTFMT            = 00000029
ICB$B_DSTOBJ            = 0000002A
ICB$B_LPRNAM            = 00000014
ICB$B_RID               = 00000092
ICB$B_RPRNAM            = 00000028
ICB$C_ACCESS            = 00000040
ICB$C_LENGTH            = 000000A3
ICB$C_RID               = 00000010
ICB$C_RPRNAM            = 00000014
ICB$T_ACCESS            = 0000003D
ICB$T_DSTDSC            = 0000002B
ICB$T_RID               = 00000093
ICB$W_DLY_FACT          = 0000000E
ICB$W_DLY_WGHT          = 00000010
ICB$W_LOCLNK            = 00000002
ICB$W_PATH              = 00000000
ICB$W_REMNOD            = 0000008D
ICB$W_RETRAN            = 0000000C
ICB$W_SEGSIZ            = 00000012
ICB$W_TIM_INACT         = 00000006
ICB$W_TIM_OCON          = 00000004
INT_B_PRX                 0000001E R      03
IRP$L_DIAGBUF           = 0000004C
IRP$L_PID               = 0000000C
IRP$L_SVAPTE            = 0000002C
IRP$L_UCB               = 0000001C
IRP$Q_NT_PRVMSK         = 00000040
JPI$_USERNAME           = 00000202
JPI_B_UNAME               00000028 R      03
JPI_ITEM_LIST             00000038 R      03
JPI_Q_IOSB                00000020 R      03
JPI_T_UNAME               0000002C R      03
LPD$B_PLVEC             = 00000028
LPD$W_PTH               = 00000020
LSB                     = 00000000
LSB$B_R_CXBCNT          = 00000028
LSB$B_R_CXBQUO          = 00000029
LSB$B_SPARE             = 0000002A
LSB$B_STS               = 0000002B
LSB$B_X_ADJ             = 0000000B
LSB$B_X_CXBACT          = 0000000D
LSB$B_X_CXBCNT          = 0000000F
LSB$B_X_CXBQUO          = 0000000E
LSB$B_X_PKTWND          = 0000000C
LSB$B_X_REQ             = 0000000A
LSB$L_CROSS             = 0000002C
LSB$L_R_CXB             = 00000020
LSB$L_R_IRP             = 0000001C
LSB$L_X_CXB             = 00000018
LSB$L_X_IRP             = 00000014
LSB$L_X_PND             = 00000010
LSB$M_BOM               = 00000020
LSB$M_EOM               = 00000040
LSB$M_LI                = 00000001
LSB$S_LSB               = 00000030
LSB$S_SPARE             = 00000004
LSB$S_STS               = 00000001
LSB$V_BOM               = 00000005
LSB$V_EOM               = 00000006
LSB$V_LI                = 00000000
LSB$V_SPARE             = 00000001
LSB$W_HAA               = 00000008
LSB$W_HAR               = 00000006
LSB$W_HAX               = 00000026
LSB$W_HNR               = 00000024
LSB$W_HXS               = 00000004
LSB$W_LNX               = 00000002
LSB$W_LUX               = 00000000
NDI_B_ACC                 0000001C R      03
NDI_PTR                   00000004 R      03
NET$AB_ACC_TAB            00000230 R      02
NET$AB_OBJTRAN            00000130 R      02
NET$AB_UPASCNUM           00000030 RG     02
NET$ALONPGD_Z             ********    X   04
NET$CONNECT               00000000 RG     04
```

```
NET$CONNECT_FAIL        ********  X   04
NET$C_ACT_TIMER       = 0000001E
NET$C_DR_SHUT         = 00000003
NET$C_EFN_ASYN        = 00000002
NET$C_EFN_WAIT        = 00000001
NET$C_IPL             = 00000008
NET$C_MAXACCFLD       = 00000027
NET$C_MAXLINNAM       = 0000000F
NET$C_MAXLNK          = 000003FF
NET$C_MAXNODNAM       = 00000006
NET$C_MAXOBJNAM       = 0000000C
NET$C_MAX_AREAS       = 0000003F
NET$C_MAX_LINES       = 00000040
NET$C_MAX_NCB         = 0000006E
NET$C_MAX_NODES       = 000003FF
NET$C_MAX_OBJ         = 000000FF
NET$C_MAX_WQE         = 00000014
NET$C_MINBUFSIZ       = 000000C0
NET$C_TID_ACT         = 00000003
NET$C_TID_RUS         = 00000001
NET$C_TID_XRT         = 00000002
NET$C_TRCTL_CEL       = 00000002
NET$C_TRCTL_OVR       = 00000005
NET$C_UTLBUFSIZ       = 00001000
NET$DEALLOCATE          ********  X   04
NET$FIND_ADJ            ********  X   04
NET$GETUTLBUF           ********  X   04
NET$GL_CNR_CRI          ********  X   04
NET$GL_CNR_NDI          ********  X   04
NET$GL_CNR_OBI          ********  X   04
NET$GL_CNR_PLI          ********  X   04
NET$GL_FLAGS            ********  X   04
NET$GL_PTR_VCB          ********  X   04
NET$GL_SAVE_IRP         ********  X   04
NET$GL_UTLBUF           ********  X   04
NET$M_MAXLNKMSK       = 000003FF
NET$M_RQIRP           = 00000020
NET$NDI_BY_ADD          ********  X   04
NET$PROC_XOB            ********  X   04
NET$TEST_REACH          ********  X   04
NETUPD$_CRELNK        = 00000007
NETUPD$_TEST_ADJ      = 0000000F
NFB$C_CRI_NAM         = 04020041
NFB$C_NDI_ACC         = 02010020
NFB$C_NDI_ADD         = 02010012
NFB$C_NDI_LOO         = 02000002
NFB$C_NDI_NAC         = 02020052
NFB$C_NDI_NLI         = 0202004C
NFB$C_NDI_NNA         = 02020043
NFB$C_NDI_NPW         = 02020053
NFB$C_NDI_NUS         = 02020051
NFB$C_NDI_PAC         = 0202004F
NFB$C_NDI_PPW         = 02020050
NFB$C_NDI_PUS         = 0202004E
NFB$C_OBI_HPR         = 03010011
NFB$C_OBI_LPR         = 03010010
NFB$C_OBI_NAM         = 03020044
NFB$C_OBI_NUM         = 03010014
NFB$C_OBI_PRX         = 03010016
NFB$C_OP_EQL          = 00000000
NFB$C_PLI_BFS         = 05010027
NFB$C_PLI_PLVEC       = 05010020
NMA$C_ACES_BOTH       = 00000003
NMA$C_ACES_INCO       = 00000001
NMA$C_ACES_NONE       = 00000000
NMA$C_ACES_OUTG       = 00000002
NONPRV_TAB              00000010 R    02
NSP$$$_QUAL_ACK       = 00000000
NSP$$$_QUAL_ALTFLW    = 00000000
NSP$$$_QUAL_DATA      = 00000000
NSP$$$_QUAL_FLW       = 00000000
NSP$$$_QUAL_INF       = 00000000
NSP$$$_QUAL_MSG       = 00000000
NSP$$$_QUAL_SRV       = 00000000
NSP$C_EXT_LNK         = 0000001E
NSP$C_FLW_DATA        = 00000000
NSP$C_FLW_INT         = 00000001
NSP$C_FLW_NOP         = 00000000
NSP$C_FLW_XOFF        = 00000001
NSP$C_FLW_XON         = 00000002
NSP$C_HSZ_ACK         = 00000007
NSP$C_HSZ_CA          = 00000003
NSP$C_HSZ_CC          = 00000064
NSP$C_HSZ_CD          = 000000F0
NSP$C_HSZ_CI          = 000000F0
NSP$C_HSZ_DATA        = 00000009
NSP$C_HSZ_DC          = 00000016
NSP$C_HSZ_DI          = 00000016
NSP$C_HSZ_INT         = 00000009
NSP$C_HSZ_LS          = 00000009
NSP$C_INF_V31         = 00000001
NSP$C_INF_V32         = 00000000
NSP$C_INF_V33         = 00000002
NSP$C_MAXADR          = 00000009
NSP$C_MSG_CA          = 00000024
NSP$C_MSG_CC          = 00000028
NSP$C_MSG_CI          = 00000018
NSP$C_MSG_DATA        = 00000000
NSP$C_MSG_DC          = 00000048
NSP$C_MSG_DI          = 00000038
NSP$C_MSG_DTACK       = 00000004
NSP$C_MSG_INT         = 00000030
NSP$C_MSG_LIACK       = 00000014
NSP$C_MSG_LS          = 00000010
NSP$C_SRV_MFC         = 00000002
NSP$C_SRV_NFC         = 00000000
NSP$C_SRV_REQ         = 00000001
NSP$C_SRV_SFC         = 00000001
NSP$M_ACK_NAK         = 00001000
NSP$M_ACK_NUM         = 00000FFF
NSP$M_ACK_VALID       = 00008000
NSP$M_DATA_BOM        = 00000020
NSP$M_DATA_EOM        = 00000040
NSP$M_DATA_OVFW       = 00000080
```

```
NSP$M_FLW_CHAN          = 0000000C
NSP$M_FLW_DRV           = 000000F0
NSP$M_FLW_INT           = 00000020
NSP$M_FLW_INUSE         = 00000010
NSP$M_FLW_LISUB         = 00000004
NSP$M_FLW_MODE          = 00000003
NSP$M_FLW_SP1           = 00000008
NSP$M_FLW_SP2           = 00000040
NSP$M_FLW_SP3           = 00000080
NSP$M_FLW_XOFF          = 00000001
NSP$M_FLW_XON           = 00000002
NSP$M_INF_VER           = 00000003
NSP$M_MSG_INT           = 00000020
NSP$M_MSG_LI            = 00000010
NSP$M_SRV_01            = 00000003
NSP$M_SRV_EXT           = 00000080
NSP$M_SRV_FLW           = 0000000C
NSP$M_SRV_REQ           = 000000F3
NSP$M_SRV_SP1           = 00000070
NSP$R_QUAL              = 00000000
NSP$S_ACK_NUM           = 0000000C
NSP$S_ACK_SP2           = 00000002
NSP$S_DATA_SP           = 00000005
NSP$S_FLW_CHAN          = 00000002
NSP$S_FLW_DRV           = 00000004
NSP$S_FLW_MODE          = 00000002
NSP$S_INF_VER           = 00000002
NSP$S_MSG_SP1           = 00000004
NSP$S_NSPMSG            = 00000005
NSP$S_QUAL              = 00000005
NSP$S_QUAL_ACK          = 00000002
NSP$S_QUAL_ALTFLW       = 00000001
NSP$S_QUAL_DATA         = 00000001
NSP$S_QUAL_FLW          = 00000001
NSP$S_QUAL_INF          = 00000001
NSP$S_QUAL_MSG          = 00000005
NSP$S_QUAL_SRV          = 00000001
NSP$S_SRV_01            = 00000002
NSP$S_SRV_FLW           = 00000002
NSP$S_SRV_SP1           = 00000003
NSP$V_ACK_NAK           = 0000000C
NSP$V_ACK_NUM           = 00000000
NSP$V_ACK_SP2           = 0000000D
NSP$V_ACK_VALID         = 0000000F
NSP$V_DATA_BOM          = 00000005
NSP$V_DATA_EOM          = 00000006
NSP$V_DATA_OVFW         = 00000007
NSP$V_DATA_SP           = 00000000
NSP$V_FLW_CHAN          = 00000002
NSP$V_FLW_DRV           = 00000004
NSP$V_FLW_INT           = 00000005
NSP$V_FLW_INUSE         = 00000004
NSP$V_FLW_LISUB         = 00000002
NSP$V_FLW_MODE          = 00000000
NSP$V_FLW_SP1           = 00000003
NSP$V_FLW_SP2           = 00000006
NSP$V_FLW_SP3           = 00000007
NSP$V_FLW_XOFF          = 00000000
NSP$V_FLW_XON           = 00000001
NSP$V_INF_VER           = 00000000
NSP$V_MSG_INT           = 00000005
NSP$V_MSG_LI            = 00000004
NSP$V_MSG_SP1           = 00000000
NSP$V_SRV_01            = 00000000
NSP$V_SRV_EXT           = 00000007
NSP$V_SRV_FLW           = 00000002
NSP$V_SRV_SP1           = 00000004
NSP$W_DSTLNK            = 00000001
NSP$W_SRCLNK            = 00000003
OBI_B_PRX                 0000001D R      03
OBI_PTR                   00000008 R      03
OBJ_Q_DESC                0000000C R      03
PRS_ACCESS                000003E0 R      04
PRS_END                   00000593 R      04
PRS_NCB                   000001C2 R      04
PRS_NODE                  00000216 R      04
PRS_OBJECT                00000433 R      04
PRV$V_NETMBX            = 00000014
PRV$V_OPER              = 00000012
PRV$V_TMPMBX            = 0000000F
PRV_TAB                   00000000 R      02
RCB$B_ECL_DAC           = 00000066
RCB$B_ECL_DFA           = 00C00064
RCB$B_ECL_DPX           = 00000067
RCB$B_ECL_DWE           = 00000065
RCB$B_ECL_RFA           = 00000063
RCB$B_ETY               = 0000008A
RCB$B_HOMEAREA          = 0000008B
RCB$B_STI               = 00000061
RCB$W_ADDR              = 0000000E
RCB$W_DRT               = 000000AA
RCB$W_ECLSEGSIZ         = 0000007C
RCB$W_TIM_CNO           = 00000078
RCB$W_TIM_IAT           = 00000074
SCAN_BLANKS               000007C3 R      04
SIZ...                  = 00000001
SPACE                   = 00000020
SS$_DEVOFFLINE            ********    X   04
SS$_INVLOGIN              ********    X   04
SS$_IVDEVNAM              ********    X   04
SS$_NOLINKS               ********    X   04
SS$_NORMAL                ********    X   04
SS$_NOSUCHNODE            ********    X   04
SS$_NOSUCHOBJ             ********    X   04
SS$_SHUT                  ********    X   04
SS$_TOOMUCHDATA           ********    X   04
SYS$GETJPI                ********   GX   04
SYS$WAITFR                ********   GX   04
TAB                     = 00000009
TR$C_MAXHDR             = 0000001C
TR$C_NI_ALLEND1         = 040000AB
TR$C_NI_ALLEND2         = 00000000
TR$C_NI_ALLROU1         = 030000AB
TR$C_NI_ALLROU2         = 00000000
```

```
TR$C_NI_PREFIX                  = 000400AA
TR$C_NI_PROT                    = 00000360
TR$C_PRI_ECL                    = 0000001F
TR$C_PRI_RTHRU                  = 0000001F
TR3$$$_QUAL_MSG                 = 00000000
TR3$$$_QUAL_RTFLG               = 00000000
TR3$C_MSZ_DATA                  = 00000006
TR3$C_MSG_DATA                  = 00000002
TR3$C_MSG_HELLO                 = 00000005
TR3$C_MSG_INIT                  = 00000001
TR3$C_MSG_NOP2                  = 00000008
TR3$C_MSG_ROUT                  = 00000007
TR3$C_MSG_STR2                  = 00000058
TR3$C_MSG_VERF                  = 00000003
TR3$M_MSG_CTL                   = 00000001
TR3$M_MSG_RTH                   = 00000002
TR3$M_RTFLG_PH2                 = 00000040
TR3$M_RTFLG_RQR                 = 00000008
TR3$M_RTFLG_RTS                 = 00000010
TR3$R_QUAL                      = 00000000
TR3$S_QUAL                      = 00000001
TR3$S_QUAL_MSG                  = 00000001
TR3$S_QUAL_RTFLG                = 00000001
TR3$S_RTFLG_012                 = 00000003
TR3$S_TR3MSG                    = 00000001
TR3$V_MSG_CTL                   = 00000000
TR3$V_MSG_RTH                   = 00000001
TR3$V_RTFLG_012                 = 00000000
TR3$V_RTFLG_5                   = 00000005
TR3$V_RTFLG_7                   = 00000007
TR3$V_RTFLG_PH2                 = 00000006
TR3$V_RTFLG_RQR                 = 00000003
TR3$V_RTFLG_RTS                 = 00000004
TR4$$$_QUAL_ADDR                = 00000000
TR4$$$_QUAL_RTFLG               = 00000000
TR4$$$_QUAL_SCLASS              = 00000000
TR4$C_BCE_MID1                  = 040000AB
TR4$C_BCE_MID2                  = 00000000
TR4$C_BCR_MID1                  = 030000AB
TR4$C_BCR_MID2                  = 00000000
TR4$C_BCT3MULT                  = 00000008
TR4$C_END_NODE                  = 00000003
TR4$C_HIORD                     = 000400AA
TR4$C_HSZ_DATA                  = 00000015
TR4$C_MSG_BCEHEL                = 0000000D
TR4$C_MSG_BCRHEL                = 0000000B
TR4$C_MSG_LDATA                 = 00000006
TR4$C_MSG_RDATA                 = 00000002
TR4$C_PRO_TYPE                  = 00000360
TR4$C_RTR_LVL1                  = 00000002
TR4$C_RTR_LVL2                  = 00000001
TR4$C_T3MULT                    = 00000002
TR4$C_VER_HIB                   = 00000000
TR4$C_VER_LOWW                  = 00000002
TR4$M_ADDR_AREA                 = 0000FC00
TR4$M_ADDR_DEST                 = 000003FF
TR4$M_RTFLG_INI                 = 00000020
TR4$M_RTFLG_LNG                 = 00000004
TR4$M_RTFLG_RQR                 = 00000008
TR4$M_RTFLG_RTS                 = 00000010
TR4$R_QUAL                      = 00000000
TR4$S_ADDR_AREA                 = 00000006
TR4$S_ADDR_DEST                 = 0000000A
TR4$S_QUAL                      = 00000002
TR4$S_QUAL_ADDR                 = 00000002
TR4$S_QUAL_RTFLG                = 00000001
TR4$S_QUAL_SCLASS               = 00000001
TR4$S_RTFLG_01                  = 00000002
TR4$S_RTFLG_VER                 = 00000002
TR4$S_SCLASS_57                 = 00000003
TR4$S_TR4MSG                    = 00000002
TR4$V_ADDR_AREA                 = 0000000A
TR4$V_ADDR_DEST                 = 00000000
TR4$V_RTFLG_01                  = 00000000
TR4$V_RTFLG_INI                 = 00000005
TR4$V_RTFLG_LNG                 = 00000002
TR4$V_RTFLG_RQR                 = 00000003
TR4$V_RTFLG_RTS                 = 00000004
TR4$V_RTFLG_VER                 = 00000006
TR4$V_SCLASS_1                  = 00000001
TR4$V_SCLASS_57                 = 00000005
TR4$V_SCLASS_BC                 = 00000004
TR4$V_SCLASS_LS                 = 00000002
TR4$V_SCLASS_METR               = 00000000
TR4$V_SCLASS_SUBA               = 00000003
TSK_Q_DESC                        00000014 R      03
XWB                             = 00000000
XWB$B_ACCESS                    = 0000000B
XWB$B_DATA                      = 0000005B
XWB$B_FIPL                      = 0000001F
XWB$B_LOGIN                     = 000000CC
XWB$B_LPRNAM                    = 000000A4
XWB$B_PRO                       = 0000005A
XWB$B_RID                       = 0000006F
XWB$B_RPRNAM                    = 000000B8
XWB$B_SP3                       = 0000006E
XWB$B_STA                       = 0000001E
XWB$B_TYPE                      = 0000000A
XWB$B_X_FLW                     = 0000006C
XWB$B_X_FLWCNT                  = 0000006D
XWB$C_COMLNG                    = 000000A4
XWB$C_CONLNG                    = 00000112
XWB$C_DATA                      = 00000010
XWB$C_LOGIN                     = 00000040
XWB$C_LPRNAM                    = 00000014
XWB$C_NDC_LNG                   = 00000020
XWB$C_NUMSTA                    = 00000008
XWB$C_RID                       = 00000010
XWB$C_RPRNAM                    = 00000014
XWB$C_STA_CAR                   = 00000002
XWB$C_STA_CCS                   = 00000004
XWB$C_STA_CIR                   = 00000003
XWB$C_STA_CIS                   = 00000001
XWB$C_STA_CLO                   = 00000000
```

```
XWB$C_STA_DIR                   = 00C00006
XWB$C_STA_DIS                   = 000 0007
XWB$C_STA_RUN                   = 000 0005
XWB$L_DEA_IRP                   = 00( )0104
XWB$L_FPC                       = 00000020
XWB$L_FR3                       = 00000024
XWB$L_FR4                       = 00000028
XWB$L_ICB                       = 0000010C
XWB$L_IRP_ACC                   = 00000080
XWB$L_LINK                      = 0000002C
XWB$L_ORGUCB                    = 00000010
XWB$L_PID                       = 00000034
XWB$L_VCB                       = 00000030
XWB$L_WLBL                      = 00000004
XWB$L_WLFL                      = 00000000
XWB$M_FLG_BREAK                 = 00000001
XWB$M_FLG_CLO                   = 00000200
XWB$M_FLG_IAVL                  = 00001000
XWB$M_FLG_SCD                   = 00000100
XWB$M_FLG_SDACK                 = 00000008
XWB$M_FLG_SDFL                  = 00004000
XWB$M_FLG_SDT                   = 00000080
XWB$M_FLG_SIACK                 = 00000004
XWB$M_FLG_SIFL                  = 00002000
XWB$M_FLG_SLI                   = 00000010
XWB$M_FLG_TBPR                  = 00000800
XWB$M_FLG_WBP                   = 00000040
XWB$M_FLG_WBUF                  = 00000002
XWB$M_FLG_WDAT                  = 00000400
XWB$M_FLG_WHGL                  = 00000020
XWB$M_PRO_CCA                   = 00000008
XWB$M_PRO_NAR                   = 00000010
XWB$M_PRO_NFC                   = 00000001
XWB$M_PRO_PH2                   = 00000004
XWB$M_PRO_SFC                   = 00000002
XWB$M_STS_ASTPND                = 00000400
XWB$M_STS_ASTREQ                = 00000800
XWB$M_STS_CON                   = 00000010
XWB$M_STS_DIS                   = 00000008
XWB$M_STS_DTNAK                 = 00000100
XWB$M_STS_LINAK                 = 00000200
XWB$M_STS_NDC                   = 00001000
XWB$M_STS_OVF                   = 00000080
XWB$M_STS_RBP                   = 00000040
XWB$M_STS_SOL                   = 00000004
XWB$M_STS_TID                   = 00000001
XWB$M_STS_TLI                   = 00000002
XWB$M_STS_TMO                   = 00000020
XWB$Q_FORK                      = 00000014
XWB$Q_FREE_CXB                  = 00000118
XWB$R_CON_BLK                   = 000000A4
XWB$R_RUN_BLK                   = 000000A4
XWB$S                           = 00000006
XWB$S_COMLNG                    = 0000006E
XWB$S_CON_BLK                   = 0000006E
XWB$S_DATA                      = 00000010
XWB$S_DT                        = 00000030
XWB$S_FLG                       = 00000002
XWB$S_FORK                      = 00000008
XWB$S_FREE_CXB                  = 00000008
XWB$S_LI                        = 00000030
XWB$S_LOGIN                     = 0000003F
XWB$S_LPRNAM                    = 00000013
XWB$S_NDC                       = 00000020
XWB$S_PRO                       = 00000001
XWB$S_RID                       = 00000010
XWB$S_RPRNAM                    = 00000013
XWB$S_RUN_BLK                   = 00000064
XWB$S_STS                       = 00000002
XWB$S_XWB                       = 00000120
XWB$T                           = 00000112
XWB$T_DATA                      = 0000005C
XWB$T_DT                        = 000000A4
XWB$T_LI                        = 000000D4
XWB$T_LOGIN                     = 000000CD
XWB$T_LPRNAM                    = 000000A5
XWB$T_RID                       = 00000070
XWB$T_RPRNAM                    = 000000B9
XWB$V_FLG_BREAK                 = 00000000
XWB$V_FLG_CLO                   = 00000009
XWB$V_FLG_IAVL                  = 0000000C
XWB$V_FLG_SCD                   = 00000008
XWB$V_FLG_SDACK                 = 00000003
XWB$V_FLG_SDFL                  = 0000000E
XWB$V_FLG_SDT                   = 00000007
XWB$V_FLG_SIACK                 = 00000002
XWB$V_FLG_SIFL                  = 0000000D
XWB$V_FLG_SLI                   = 00000004
XWB$V_FLG_TBPR                  = 0000000B
XWB$V_FLG_WBP                   = 00000006
XWB$V_FLG_WBUF                  = 00000001
XWB$V_FLG_WDAT                  = 0000000A
XWB$V_FLG_WHGL                  = 00000005
XWB$V_PRO_CCA                   = 00000003
XWB$V_PRO_NAR                   = 00000004
XWB$V_PRO_NFC                   = 00000000
XWB$V_PRO_PH2                   = 00000002
XWB$V_PRO_SFC                   = 00000001
XWB$V_STS_ASTPND                = 0000000A
XWB$V_STS_ASTREQ                = 0000000B
XWB$V_STS_CON                   = 00000004
XWB$V_STS_DIS                   = 00000003
XWB$V_STS_DTNAK                 = 00000008
XWB$V_STS_LINAK                 = 00000009
XWB$V_STS_NDC                   = 0000000C
XWB$V_STS_OVF                   = 00000007
XWB$V_STS_RBP                   = 00000006
XWB$V_STS_SOL                   = 00000002
XWB$V_STS_TID                   = 00000000
XWB$V_STS_TLI                   = 00000001
XWB$V_STS_TMO                   = 00000005
XWB$W_CI_PATH                   = 00000110
XWB$W_DECAY                     = 0000004E
XWB$W_DLY_FACT                  = 00000056
```

```
XWB$W_DLY_WGHT                  = 00000058
XWB$W_ELAPSE                    = 0000004A
XWB$W_FLG                       = 0000001C
XWB$W_LOCLNK                    = 0000003E
XWB$W_LOCSIZ                    = 00000040
XWB$W_PATH                      = 00000038
XWB$W_PROGRESS                  = 00000052
XWB$W_REFCNT                    = 0000000C
XWB$W_REMLNK                    = 0000003C
XWB$W_REMNOD                    = 0000003A
XWB$W_REMSIZ                    = 00000042
XWB$W_RETRAN                    = 00000054
XWB$W_R_REASON                  = 00000044
XWB$W_SIZE                      = 00000008
XWB$W_STS                       = 0000000E
XWB$W_TIMER                     = 00000050
XWB$W_TIM_ID                    = 00000048
XWB$W_TIM_INACT                 = 0000004C
XWB$W_X_REASON                  = 00000046
XWB$Z_NDC                       = 00000084
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	00000000 (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
NET_PURE	00000330 (816.)	02 (2.)	NOPIC USR CON REL LCL NOSHR NOEXE RD NOWRT NOVEC LONG
NET_IMPURE	00000048 (72.)	03 (3.)	NOPIC USR CON REL LCL NOSHR NOEXE RD WRT NOVEC LONG
NET_CODE	000007D2 (2002.)	04 (4.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC BYTE

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	26	00:00:00.12	00:00:00.94
Command processing	140	00:00:01.05	00:00:04.96
Pass 1	1084	00:00:30.39	00:00:43.29
Symbol table sort	19	00:00:04.09	00:00:04.88
Pass 2	690	00:00:06.18	00:00:07.82
Symbol table output	72	00:00:00.47	00:00:00.96
Psect synopsis output	4	00:00:00.03	00:00:00.03
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	2038	00:00:42.34	00:01:02.89

```
The working set limit was 1950 pages.
173908 bytes (340 pages) of virtual memory were used to buffer the intermediate code.
There were 160 pages of symbol table space allocated to hold 2806 non-local and 87 local symbols.
1316 source lines were read in Pass 1, producing 26 object records in Pass 2.
63 pages of virtual memory were used to define 45 macros.
```

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SHRLIB]NMALIBRY.MLB;1	1
_255DUA28:[SHRLIB]EVCDEF.MLB;1	0
_255DUA28:[NETACP.OBJ]NETDRV.MLB;1	2
_255DUA28:[NETACP.OBJ]NET.MLB;1	16
_255DUA28:[SYS.OBJ]LIB.MLB;1	4
_255DUA28:[SYSLIB]STARLET.MLB;2	12
TOTALS (all libraries)	35

3030 GETS were required to define 35 macros.

There were no errors, warnings or information messages.

```
MACRO/LIS=LIS$:NETCONECT/OBJ=OBJ$:NETCONECT MSRC$:NETCONECT/UPDATE=(ENH$:NETCONECT)+EXECML$/LIB+LIB$:NET/LIB+LIB$:NETDRV/LIB+SHRLIB$
```

NETDLE
LIS

NETCONFIG
LIS

NETCONECT
LIS

NETCTLALL
LIS