B 9

```
NNN         NNN  EEEEEEEEEEEEEEE  TTTTTTTTTTTTTTT     AAAAAAAAA        CCCCCCCCCCCC  PPPPPPPPPPPP
NNN         NNN  EEEEEEEEEEEEEEE  TTTTTTTTTTTTTTT     AAAAAAAAA        CCCCCCCCCCCC  PPPPPPPPPPPP
NNN         NNN  EEEEEEEEEEEEEEE  TTTTTTTTTTTTTTT     AAAAAAAAA        CCCCCCCCCCCC  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  EEEEEEEEEEEE           TTT        AAA         AAA  CCC              PPPPPPPPPPPP
NNN   NNN   NNN  EEEEEEEEEEEE           TTT        AAA         AAA  CCC              PPPPPPPPPPPP
NNN   NNN   NNN  EEEEEEEEEEEE           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     CCCCCCCCCCCC  PPP
NNN         NNN  EEEEEEEEEEEEEEE        TTT        AAA         AAA     CCCCCCCCCCCC  PPP
NNN         NNN  EEEEEEEEEEEEEEE        TTT        AAA         AAA     CCCCCCCCCCCC  PPP
```

Ps

NE

NE

NE

NE

$R

E 11

FILEID**NETLLICNT

```
NN      NN  EEEEEEEEEE  TTTTTTTTTT  LL          LL            IIIIII      CCCCCCCC  NN      NN  TTTTTTTTTT
NN      NN  EEEEEEEEEE  TTTTTTTTTT  LL          LL            IIIIII      CCCCCCCC  NN      NN  TTTTTTTTTT
NN      NN  EE              TT      LL          LL              II      CC          NN      NN      TT
NN      NN  EE              TT      LL          LL              II      CC          NN      NN      TT
NNNN    NN  EE              TT      LL          LL              II      CC          NNNN    NN      TT
NNNN    NN  EE              TT      LL          LL              II      CC          NNNN    NN      TT
NN  NN  NN  EEEEEEEE        TT      LL          LL              II      CC          NN  NN  NN      TT
NN  NN  NN  EEEEEEEE        TT      LL          LL              II      CC          NN  NN  NN      TT
NN    NNNN  EE              TT      LL          LL              II      CC          NN    NNNN      TT
NN    NNNN  EE              TT      LL          LL              II      CC          NN    NNNN      TT
NN      NN  EE              TT      LL          LL              II      CC          NN      NN      TT          ....
NN      NN  EE              TT      LL          LL              II      CC          NN      NN      TT          ....
NN      NN  EEEEEEEEEE      TT      LLLLLLLLLL  LLLLLLLLLL    IIIIII      CCCCCCCC  NN      NN      TT          ....
NN      NN  EEEEEEEEEE      TT      LLLLLLLLLL  LLLLLLLLLL    IIIIII      CCCCCCCC  NN      NN      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          .TITLE  NETLLICNT - Counter support for nodes and logical-links
0000          .IDENT  'V04-000'
0000          .DEFAULT DISPLACEMENT,LONG
0000
0000  ;****************************************************************************
0000  ;*                                                                          *
0000  ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000  ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000  ;*  ALL RIGHTS RESERVED.                                                    *
0000  ;*                                                                          *
0000  ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000  ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000  ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000  ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000  ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000  ;*  TRANSFERRED.                                                            *
0000  ;*                                                                          *
0000  ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000  ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000  ;*  CORPORATION.                                                            *
0000  ;*                                                                          *
0000  ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000  ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000  ;*                                                                          *
0000  ;*                                                                          *
0000  ;****************************************************************************
0000  ;
0000  ; FACILITY:     NETWORK ACP
0000  ;
0000  ; ABSTRACT:     This module contains routines to maintain the Node counter
0000  ;               and logical-link counter databases.
0000  ;
0000  ;
0000  ; ENVIRONMENT:  Kernel mode
0000  ;
0000  ;       .SBTTL  HISTORY
0000  ;
0000  ; AUTHOR:       Alan D. Eldridge        13-Feb-1984
0000  ;
0000  ; MODIFIED BY:
0000  ;
0000  ;       V03-002 PPB0346         Paul Beck       9-Aug-1984  18:39
0000  ;               Supply error message for error returns from NET$ACQUIRE_NDCOU
0000  ;               Also, clear NDC interlock bit when transaction count in NDC is
0000  ;               decremented, not just when it reaches zero.
0000  ;
0000  ;       V03-001 RNG0001                 Rod Gamache     23-May-1984
0000  ;               Add NDC interlock bit to XWB_STS to indicate when the XWB
0000  ;               has succeeded in acquiring an NDCOU block.
0000  ;               Log counters when a database entry is reused.
0000  ;
```

```
          0000     ;
          0000     ;   Read and Optionally Zero Link xxx Counters
          0000     ;
          0000     ;       K        = XWB_x
          0000     ;       XWB_x    = 0                                 ; iff zeroing
          0000     ;       LLI_rt   = LLI_rt + K                        ; iff zeroing
          0000     ;       Return (K)
          0000     ;
          0000
          0000     ;
          0000     ;   Read and Optionally Zero Node nnn Counters
          0000     ;
          0000     ;       SK  = NDC
          0000     ;       NDC = 0                                      ; iff zeroing
          0000     ;       For all logical-links to node nnn do
          0000     ;          SK      = SK + XWB_x + LLI_rt - LLI_lz
          0000     ;          LLI_lz =      XWB_x + LLI_rt              ; iff zeroing
          0000     ;          End
          0000     ;       Return (SK)
          0000     ;
          0000
          0000     ;
          0000     ; On IO$_DEACCESS (AR = Accounting record)
          0000     ;
          0000     ;       AR  = LLI_rt + XWB_x
          0000     ;       NDC = NDC + AR
          0000     ;
          0000
          0000             $CNFDEF
          0000             $EVCDEF
          0000             $LLIDEF
          0000             $NFBDEF
          0000             $WQEDEF
          0000             $XWBDEF
          0000             $NETSYMDEF
          0000
          0000
          0000
00000024  0000             CNF = CNF$C_LENGTH                     ; Short name for readabilty
          0000
00000190  0000             LOGBUF_LEN = 400                       ; Length buffer for logging ctrs
          0000
          0000     $DEFINI NDCOU
          0000
          0000     $DEF    NDCOU$Q_LINKAGE .BLKQ 1               ; Queue linkage
          0008     $DEF    NDCOU$W_SIZE    .BLKW 1               ; Structure size
          000A     $DEF    NDCOU$B_TYPE    .BLKB 1               ; Structure type
          000B     $DEF    NDCOU$B_STS     .BLKB 1               ; Status flags
          000C     $DEF    NDCOU$L_LINK    .BLKL 1               ; Hash Table linkage
          0010     $DEF    NDCOU$W_PNA     .BLKW 1               ; Remote node address
          0012     $DEF    NDCOU$W_REFCNT  .BLKW 1               ; Number of referencers
          0014     $DEF    NDCOU$Z_NDC     .BLKB NDC$C_LENGTH    ; Counter block
00000030  0030             NDCOU$C_LENGTH  = .                   ; Total structure length
          0030
          0030     $DEFEND NDCOU
          0000
```

```
                         00000000         .PSECT  NET_LOCK_IMPURE, WRT,NOEXE,LONG ; Goto impure locked area Psect
                             0000
                     00000030  0000  NDCOU_C_SIZE      = <NDCOU$C_LENGTH + ^X<F>> & ^C^X<F>  ; Round up NDCOU size
                             0000
                     00000030  0000  TEMP_Z_NDC:          .BLKB       NDCOU_C_SIZE            ; Temporary NDCOU
                             0030
                         00000000         .PSECT  NET_IMPURE, WRT,NOEXE,QUAD      ; Goto impure area Psect
                             0000         .ALIGN  QUAD
                             0000  ;
                             0000  ;
                             0000  ;    Define space for NDCOU blocks.  Allocate the blocks (via NET$INIT_NDCOU)
                             0000  ;    during NETACP initialization
                             0000  ;
                             0000  ;
                     00000200  0000  NDCOU_C_BLOCKS    =  512                                 ; Number of NDCOU blocks
                             0000
                             0000  ;
                             0000  ;
                             0000  ;    Setup 'idle' and 'inactive' NDCOU queue headers.  The 'idle queue' contains
                             0000  ;    absolutely unused NDCOU's.  The 'inactive queue' contains NDCOU with a zero
                             0000  ;    reference count but with an assigned remote node address and which are
                             0000  ;    linked into the Hash Table
                             0000  ;
                             0000  ;
                    00000000' 0000  NET$Q_NDCOU_IDLE:   .ADDRESS    NET$Q_NDCOU_IDLE        ; Idle NDCOU's
                    00000000' 0004                      .ADDRESS    NET$Q_NDCOU_IDLE        ;
                    00000008' 0008  NET$Q_NDCOU_INACT:  .ADDRESS    NET$Q_NDCOU_INACT       ; Inactive NDCOU's
                    00000008' 000C                      .ADDRESS    NET$Q_NDCOU_INACT       ;
                             0010
                             0010  ;
                             0010  ;
                             0010  ;    Setup NDCOU Hash Table -- contains no entries at the start.
                             0010  ;
                             0010  ;
                     00000080  0010  HASH_C_LNG        =  128                                 ; Length of table
                     00000000  0010  HASH_V_LNG        =    0                                 ; Parameters to trim
                     00000007  0010  HASH_S_LNG        =    7                                 ;  the Hash Table index
                             0010
                             0010  _VIELD  COU,0,<-                                         ; Counter calling interface
                             0010          <ZERO,,M>,-                                      ; Set if zero requested
                             0010          <HIGHIPL,,M>,-                                   ; Set if high IPL needed
                             0010          <,6>,-                                           ; UNDEFINED
                             0010          >
                             0010
00000000'00000000'00000000'00000000'  0010  NET$GZ_HASHT_NDCOU::   .LONG    0 [HASH_C_LNG]         ; Initialize Hash Table
00000000'00000000'00000000'00000000'  0020
00000000'00000000'00000000'00000000'  0030
00000000'00000000'00000000'00000000'  0040
00000000'00000000'00000000'00000000'  0050
00000000'00000000'00000000'00000000'  0060
00000000'00000000'00000000'00000000'  0070
00000000'00000000'00000000'00000000'  0080
00000000'00000000'00000000'00000000'  0090
00000000'00000000'00000000'00000000'  00A0
00000000'00000000'00000000'00000000'  00B0
00000000'00000000'00000000'00000000'  00C0
00000000'00000000'00000000'00000000'  00D0
```

```
00000000'00000000'00000000'00000000'  00E0
00000000'00000000'00000000'00000000'  00F0
00000000'00000000'00000000'00000000'  0100
00000000'00000000'00000000'00000000'  0110
00000000'00000000'00000000'00000000'  0120
00000000'00000000'00000000'00000000'  0130
00000000'00000000'00000000'00000000'  0140
00000000'00000000'00000000'00000000'  0150
00000000'00000000'00000000'00000000'  0160
00000000'00000000'00000000'00000000'  0170
00000000'00000000'00000000'00000000'  0180
00000000'00000000'00000000'00000000'  0190
00000000'00000000'00000000'00000000'  01A0
00000000'00000000'00000000'00000000'  01B0
00000000'00000000'00000000'00000000'  01C0
00000000'00000000'00000000'00000000'  01D0
00000000'00000000'00000000'00000000'  01E0
00000000'00000000'00000000'00000000'  01F0
00000000'00000000'00000000'00000000'  0200
                                      0210
                                      0210  .MACRO  HASH_NODE_ADDRESS  PARAM                          ; Find Hash Table addr
                                      0210
                                      0210          MOVW    PARAM,-(SP)                               ; Push node address
                                      0210          MOVZBL  (SP)+,PARAM                               ; Recover low order
                                      0210          INCB    (SP)                                      ; Make hi order nonzero
                                      0210                                                            ; most of the time
                                      0210          MULB    (SP)+,PARAM                               ; Use product
                                      0210          EXTZV   #HASH_V_LNG,#HASH_S_LNG,PARAM,PARAM       ; Trim to legal index
                                      0210          MOVAL   NET$GZ_HASHT_NDCOU[PARAM],PARAM           ; Calc. table address
                                      0210
                                      0210  .ENDM   HASH_NODE_ADDRESS
                                      0210
                                      0210  .MACRO  ADDCOU  offset,base1,base2,len=L,?L               ; Add counters
                                      0210
                                      0210          ADD'len NDC$'len'_'offset'('base1'),NDC$'len'_'offset'('base2')
                                      0210          BCC     L                                         ; Br if no carry
                                      0210          MNEG'len        #1,NDC$'len'_'offset'('base2')
                                      0210  L:
                                      0210  .ENDM   ADDCOU
                                      0210
                                      0210  .MACRO  SUBCOU  offset,base1,base2,len=L,?L               ; Subtract counters
                                      0210
                                      0210          SUB'len NDC$'len'_'offset'('base1'),NDC$'len'_'offset'('base2')
                                      0210          BCC     L                                         ; Br if no carry
                                      0210          MNEG'len        #1,NDC$'len'_'offset'('base2')
                                      0210  L:
                                      0210  .ENDM   SUBCOU
                                      0210
                                      0210
                                  00000000          .PSECT  NET_CODE, NOWRT,EXE                       ; Goto code PSECT
                                      0000
```

```
                                 0000          .SBTTL  NET$INIT_NDCOU          - Initialize NDC queues
                                 0000     ;+
                                 0000     ;
                                 0000     ;  This routine is called once when NETACP is started.  It fills the
                                 0000     ;  'idle' queue with NDCOU sized blocks
                                 0000     ;
                                 0000     ;  INPUTS:       None
                                 0000     ;
                                 0000     ;  OUTPUTS:      R0      Status returned from NET$ALLOCATE
                                 0000     ;
                                 0000     ;                All other registers are preserverd.
                                 0000     ;
                                 0000     ;-
                                 0000  NET$INIT_NDCOU::                        ; Initialize NDC queues
                     06    BB    0000          PUSHR   #^M<R1,R2>              ; Save regs
                                 0002                                          ;
      51   00006000 8F    D0    0002          MOVL    #NDCOU_C_BLOCKS -       ; Get length needed
                                 0009                  *NDCOU_C_SIZE,R1        ;
           00000000'EF    16    0009          JSB     NET$ALLOCATE            ; Go get the storage
                  17 50    E9    000F          BLBC    R0,30$                  ; If LBC, error
      50   00000200 8F    D0    0012          MOVL    #NDCOU_C_BLOCKS,R0      ; Number of NDCOU's
   00000004'FF    62    0E    0019  10$:    INSQUE  (R2),@NET$Q_NDCOU_IDLE+4; Insert it on the queue
                  52  30    C0    0020          ADDL    #NDCOU_C_SIZE,R2        ; Advance to next NDCOU
                  F3 50    F5    0023          SOBGTR  R0,10$                  ; Loop
                  50  01    D0    0026          MOVL    #1,R0                   ; Say "success"
                                 0029                                          ;
                     06    BA    0029  30$:    POPR    #^M<R1,R2>              ; Restore regs
                           05    002B          RSB                             ; Done
                                 002C
```

```
                                  002C        .SBTTL  NET$ACQUIRE_NDCOU       - Acquire Node Counter block
                                  002C  ;+
                                  002C  ;
                                  002C  ;  This routine is called to gain access to a node counter block.  For instance,
                                  002C  ;  when an XWB (logical-link control block) is created.
                                  002C  ;
                                  002C  ;  If the block does not yet exist for that node, one is removed from the 'idle
                                  002C  ;  NDCOU' queue.  It is initialized and linked into the Hash Table.
                                  002C  ;
                                  002C  ;  If the idle queue is empty, an NDCOU is removed from the front of the
                                  002C  ;  'inactive NDCOU' queue.  It is removed from the Hash Table and its contents
                                  002C  ;  are logged ("database re-used event").  It is then zeroed and linked into the
                                  002C  ;  Hash Table under its new node address.
                                  002C  ;
                                  002C  ;
                                  002C  ;  INPUTS:       R3      XWB address
                                  002C  ;                R1,R0   Scratch
                                  002C  ;
                                  002C  ;  OUTPUTS:      R1      Garbage  (Actually NDCOU ptr, but higher levels should
                                  002C  ;                                 never need to know that.)
                                  002C  ;                R0      Status (LBS/LBC)
                                  002C  ;
                                  002C  ;                All other registers are preserved.
                                  002C  ;
                                  002C  ;-
                                  002C  NET$ACQUIRE_NDCOU::                                     ; Acquire NDCOU block
             014C 8F        BB    002C          PUSHR   #^M<R2,R3,R6,R8>                        ; Save regs
                                  0030
             56   53        D0    0030          MOVL    R3,R6                                   ; Copy XWB address
          58    3A A6       3C    0033          MOVZWL  XWB$W_REMNOD(R6),R8                     ; Get the remote node address
                                  0037                                                          ;
              017D          30    0037          BSBW    NET$LOOKUP_NDCOU                        ; Find the NDCOU block
            0D 50           E9    003A          BLBC    R0,30$                                  ; If LBC, not in hash table
             51   52        D0    003D          MOVL    R2,R1                                   ; Transfer NDCOU ptr to R1
                                  0040          ;
                                  0040          ;
                                  0040          ;       NDCOU found in Hash Table.  If reference count is 0, then the
                                  0040          ;       NDCOU is also in the 'inactive NDCOU' queue -- remove it.
                                  0040          ;
                                  0040          ;
              12 A1         B5    0040          TSTW    NDCOU$W_REFCNT(R1)                      ; In 'inactive' queue ?
                 51         12    0043          BNEQ    90$                                     ; If NEQ, no
             51   61        0F    0045          REMQUE  (R1),R1                                 ; Else, remove it from queue
                 4C         11    0048          BRB     90$                                     ; Take common exit
                                  004A  30$:    ;
                                  004A          ;
                                  004A          ;       Find an unused NDCOU block and link it into the Hash Table.
                                  004A          ;
                                  004A          ;       If there is one on the 'idle' queue, then remove it and  use it.
                                  004A          ;       Otherwise, if there is one on the 'inactive' queue, then remove
                                  004A          ;       it, log it's contents, unhook it from the Hash Table, and use it.
                                  004A          ;
                                  004A          ;
     50     00000000'8F     D0    004A          MOVL    #SS$_INSFMEM,R0                         ; Anticipate no blocks
     51     00000000'FF     0F    0051          REMQUE  @NET$Q_NDCOU_IDLE,R1                    ; Get an idle block
                     18     1C    0058          BVC     50$                                     ; If VC, got one
     51     00000008'FF     0F    005A          REMQUE  @NET$Q_NDCOU_INACT,R1                   ; Get the last recently used one
```

```
                   3E  1D  0061          BVS     100$                               ; If VC none, return error
                00D4'  30  0063          BSBW    LOG_NDCOU                          ; Log the contents
           52   10 A1  3C  0066          MOVZWL  NDCOU$W_PNA(R1),R2                 ; Get node address
                 0117  30  006A          BSBW    GET_HASH_ADDR                      ; Get Hash Table address
     0C A3      0C A1  D0  006D          MOVL    NDCOU$L_LINK(R1),NDCOU$L_LINK(R3)  ; Remove it from list
                           0072     50$:  ;
                           0072           ;
                           0072           ;      Initialize the NDCOU block and link it into the Hash Table
                           0072           ;
                           0072           ;
                   3A  BB  0072          PUSHR   #^M<R1,R3,R4,R5>                   ; Save regs
61    30    00     6E  00  0074          MOVC5   #0,(SP),#0,#NDCOU$C_LENGTH,(R1)    ; Zero the block
                       2C
                   3A  BA  0079          POPR    #^M<R1,R3,R4,R5>                   ; Restore regs
                           007C                                                     ;
        08 A1      30  B0  007C          MOVW    #NDCOU$C_LENGTH,NDCOU$W_SIZE(R1)   ; Setup structure size
        10 A1      58  B0  0080          MOVW    R8,NDCOU$W_PNA(R1)                 ; Setup the node address
   14 A1    00000000'GF  D0  0084        MOVL    G^EXE$GL_ABSTIM,NDCOU$Z_NDC -      ; Initialize time since
                           008C                  +NDC$L_ABS_TIM(R1)                 ;     last zeroed
           52      58  D0  008C          MOVL    R8,R2                              ; Prepare for subr call
                 00F2  30  008F          BSBW    GET_HASH_ADDR                      ; Get the table address
        0C A3      51  D0  0092          MOVL    R1,NDCOU$L_LINK(R3)                ; Link new NDCOU into list
                           0096     90$:  ;
                           0096           ;
                           0096           ;      Acquire NDCOU by bumping its reference count.
                           0096           ;
                           0096           ;
                           0096          SETBIT  XWB$V_STS_NDC,XWB$W_STS(R6)        ; Indicate we have our NDCOU
              12 A1    B6  009B          INCW    NDCOU$W_REFCNT(R1)                 ; Account for new reference
           50      01  D0  009E          MOVL    #1,R0                              ; Say "success"
                           00A1                                                     ;
          014C 8F      BA  00A1    100$:  POPR    #^M<R2,R3,R6,R8>                  ; Restore regs
                       05  00A5          RSB
                           00A6
```

```
                                00A6          .SBTTL  NET$RELEASE_NDCOU       - Release claim on NDCOU block
                                00A6     ;+
                                00A6     ;  This routine is called to cancel a reference to an NDCOU block, e.g. when
                                00A6     ;  an XWB (logical-link control block) is being deallocated.
                                00A6     ;
                                00A6     ;  Decrement the NDCOU reference count.  If it goes to zero, insert the NDCOU
                                00A6     ;  on back of the "inactive" queue.
                                00A6     ;
                                00A6     ;
                                00A6     ;  INPUTS:       R3      XWB address
                                00A6     ;                R0      Scratch
                                00A6     ;
                                00A6     ;  OUTPUTS:      R0      Status (always LBS for now, but don't plan on it)
                                00A6     ;
                                00A6     ;                        All other registers are preserved.
                                00A6     ;
                                00A6     ;-
                                00A6     NET$RELEASE_NDCOU::                           ; Release claim on NDCOU block
                 0106 8F   BB   00A6            PUSHR   #^M<R1,R2,R8>                  ; Save regs
                                00AA                                                   ;
                       0C  E5   00AA            BBCC    #XWB$V_STS_NDC,-               ; Leave if NDCOU block not present
                  18 0E A3      00AC                     XWB$W_STS(R3),100$            ; ... else clear flag in XWB
              58    3A A3  3C   00AF            MOVZWL  XWB$W_REMNOD(R3),R8            ; Get remote node address
                     0101  30   00B3            BSBW    NET$LOOKUP_NDCOU               ; Find the NDCOU block
                    13 50  E9   00B6            BLBC    R0,200$                        ; If LBC, not there
                    12 A2  B7   00B9            DECW    NDCOU$W_REFCNT(R2)             ; One less referencer
                       0E  19   00BC            BLSS    200$                           ; If LSS, bug
                       07  12   00BE            BNEQ    100$                           ; If NEQ, we're done
        0000000C'FF    62  0E   00C0            INSQUE  (R2),@NET$Q_NDCOU_INACT+4  ; Else, queue it to end of idle queue
                                00C7                                                   ;
                 0106 8F   BA   00C7     100$:  POPR    #^M<R1,R2,R8>                  ; Restore regs
                           05   00CB            RSB                                    ; Done
                                00CC
                                00CC     200$:  BUG_CHECK  NETNOSTATE,FATAL            ; One of various bugs
                                00D0
```

```
                      00D0        .SBTTL  NET$FLUSH_LLI_CNT        - Flush logical-link counters
                      00D0  ;+
                      00D0  ;
                      00D0  ;  This routine is called when a logical-link is being deleted from the
                      00D0  ;  database.  It's total counters since creation are captured and added to
                      00D0  ;  the associated NDCOU block.  These same counters are returned for with
                      00D0  ;  the expectation that the caller may write them to some account log.
                      00D0  ;
                      00D0  ;  Basically, the following algorithm is run:
                      00D0  ;
                      00D0  ;       Begin
                      00D0  ;       AR  = LLI_rt + XWB_x
                      00D0  ;       NDC = NDC + AR
                      00D0  ;       Return (AR)
                      00D0  ;       End
                      00D0  ;
                      00D0  ;
                      00D0  ;  INPUTS:      R10     LLI CNF pointe'
                      00D0  ;               R6      Non-pageable block to receive counters in NDC format
                      00D0  ;               R5      XWB pointer
                      00D0  ;               R4      1 => Zero XWB counters
                      00D0  ;                       0 => Don't zero XWB counters
                      00D0  ;               R3-R0   Scratch
                      00D0  ;
                      00D0  ;  OUTPUTS:     R8,R3-R0   Garbage
                      00D0  ;
                      00D0  ;               All other registers are preserved.
                      00D0  ;
                      00D0  ;-
                      00D0 NET$FLUSH_LLI_CNT::                      ; Flush LLI/XWB counters to NDCOU
         0C     E1    00D0        BBC     #XWB$V_STS_NDC,-        ; Br if no NDCOU block was acquired
    17 0E A5          00D2                XWB$W_STS(R5),100$
 58   3A A5     3C    00D5        MOVZWL  XWB$W_REMNOD(R5),R8     ; Get remote node address
                      00D9        ;
                      00D9        ;
                      00D9        ;       AR  = LLI_rt + XWB_x
                      00D9        ;
                      00D9        ;
         16     10    00D9        BSBB    NET$READ_LLI_CNT        ; Fill R6 block with LLI_rt + XWB
                      00DB        ;
                      00DB        ;
                      00DB        ;       NDC = NDC + AR
                      00DB        ;
                      00DB        ;
       00D9     30    00DB        BSBW    NET$LOOKUP_NDCOU        ; Get NDCOU block for this node
      0C 50     E9    00DE        BLBC    R0,200$                 ; If LBC, bug
   52   14      C0    00E1        ADDL    #NDCOU$Z_NDC,R2         ; Setup destination block
   51   56      D0    00E4        MOVL    R6,R1                   ; Setup source NDC block
        50      D4    00E7        CLRL    R0                      ; Say "don't zero, don't use high IPL"
      FF54'     30    00E9        BSBW    ADD_NDC                 ; Add counters to NDCOU block
        05            00EC 100$:  RSB                             ; Done
                      00ED
                      00ED 200$:  BUG_CHECK  NETNOSTATE,FATAL
                      00F1
```

```
                    00F1          .SBTTL  NET$READ_LLI_CNT        - Read logical-link counters
                    00F1     ;+
                    00F1     ;
                    00F1     ; This routine is called to read, and optionally clear, the counters for a
                    00F1     ; single logical-link.   The XWB contains the copy of the counters since the
                    00F1     ; last time they were zeroed.  Whenever the XWB counters are zeroed, they are
                    00F1     ; first added to the LLI "running total" counter block so that the information
                    00F1     ; is not lost either for accounting purposes or as part of the Node counters.
                    00F1     ;
                    00F1     ; Basically, the following algorithm is run:
                    00F1     ;
                    00F1     ;            Begin
                    00F1     ;            K          = XWB_x
                    00F1     ;            XWB_x      = 0                         ; iff zeroing
                    00F1     ;            LLI_rt     = LLI_rt + K                ; iff zeroing
                    00F1     ;            Return (K)
                    00F1     ;            End
                    00F1     ;
                    00F1     ;
                    00F1     ;
                    00F1     ; INPUTS:         R10     LLI CNF pointer
                    00F1     ;                 R6      Non-pageable block to receive counters in NDC format
                    00F1     ;                 R5      XWB pointer
                    00F1     ;                 R4      1 => Zero XWB counters
                    00F1     ;                         0 => Don't zero XWB counters
                    00F1     ;                 R3-R0   Scratch
                    00F1     ;
                    00F1     ; OUTPUTS:        R3-R0   Garbage
                    00F1     ;
                    00F1     ;                 All other registers are preserved.
                    00F1     ;
                    00F1     ;
                    00F1     ;-
                    00F1     NET$READ_LLI_CNT::                          ; Read logical-link counters
     52   56   D0   00F1             MOVL    R6,R2                       ; Point to destination NDC
51   0084 C5   9E   00F4             MOVAB   XWB$Z_NDC(R5),R1            ; Point to source NDC
50   02   54   89   00F9             BISB3   R4,#COU_M_HIGHIPL,R0        ; Merge "zero" flag with "high IPL" flag
                    00FD             ;
                    00FD             ;
                    00FD             ;       K     = XWB_x
                    00FD             ;       XWB_x = 0                  (if zeroing)
                    00FD             ;
                    00FD             ;
          FF00'  30 00FD             BSBW    COPY_NDC                    ; Copy source to destination
        0C 54    E9 0100             BLBC    R4,100$                     ; If LBC, XWB NDC wasn't zeroed
                    0103             ;
                    0103             ;
                    0103             ;       LLI_rt  = LLI_rt + K
                    0103             ;
                    0103             ;
     52   2C AA  9E 0103             MOVAB   CNF+LLI$Z_NDC_RT(R10),R2; Point to 'running total' NDC
       51   56   D0 0107             MOVL    R6,R1                       ; Former XWB counter data is source NDC
            50   D4 010A             CLRL    R0                          ; Clear "zero" and "high IPL" flags
          FF31'  30 010C             BSBW    ADD_NDC                     ; Update output NDC
       50   01   D0 010F     100$:   MOVL    #1,R0                       ; Always successful
                 05 0112             RSB                                 ; Done
                    0113
```

```
                        0113          .SBTTL  NET$READ_NDI_CNT        - Read node counters
                        0113     ;+
                        0113     ;
                        0113     ;  This routine is called to calculate, and optionally zero the traffic counters
                        0113     ;  for a given remote node.  The expression (XWB_x + LLI_rt) always rep-esents
                        0113     ;  the total counters on a given link.  Thus, if a reference counter block
                        0113     ;  representing this total is maintained and updated everytime the node counters
                        0113     ;  are zeroed, then it is possible to "zero" the counters with respect to the
                        0113     ;  node counters without modifying the contents of the XWB.
                        0113     ;
                        0113     ;  Note that we cannot modify the XWB counter block in any way since that
                        0113     ;  counter block is used for the Logical-link counters.
                        0113     ;
                        0113     ;  Basically, the following algorithm is run:
                        0113     ;
                        0113     ;               Begin
                        0113     ;               SK  = NDC
                        0113     ;               NDC = 0                                 ; iff zeroing
                        0113     ;               For all logical-links to node nnn do
                        0113     ;                   Begin
                        0113     ;                   SK      = SK + XWB_x + LLI_rt - LLI_lz
                        0113     ;                   LLI_lz =       XWB_x + LLI_rt           ; iff zeroing
                        0113     ;                   End
                        0113     ;               Return (SK)
                        0113     ;               End
                        0113     ;
                        0113     ;
                        0113     ;
                        0113     ;  INPUTS:          R8      Remote node address
                        0113     ;                   R6      Non-pageable block to receive counters in NDC format
                        0113     ;                   R5      Scratch
                        0113     ;                   R4      1 => Zero XWB counters
                        0113     ;                           0 => Don't zero XWB counters
                        0113     ;                   R3-R0   Scratch
                        0113     ;
                        0113     ;  OUTPUTS:         R3-R1   Garbage
                        0113     ;                   R0      Low bit set if successful
                        0113     ;                           Low bit clear if no counter block was found
                        0113     ;
                        0113     ;                   All other registers are preserved.
                        0113     ;
                        0113     ;
                        0113     ;-
                        0113     NET$READ_NDI_CNT::                                   ; Read node counters
      0E00 8F   BB      0113             PUSHR   #^M<R9,R10,R11>                      ; Save regs
                        0117             ;
                        0117             ;
                        0117             ;       Locate associated NDCOU.  Copy and optionally zero it to
                        0117             ;       target NDCOU.
                        0117             ;
                        0117             ;
         009D   30      0117             BSBW    NET$LOOKUP_NDCOU                     ; Get NDCOU for this node
       62 50    E9      011A             BLBC    R0,200$                              ; If LBC, there is none
   51   14 A2   9E      011D             MOVAB   NDCOU$Z_NDC(R2),R1                   ; Point to source NDC
     52    56   D0      0121             MOVL    R6,R2                                ; Point to destination NDC
     50    54   D0      0124             MOVL    R4,R0                                ; Setup "zero NDC" flag
        FED6'   30      0127             BSBW    COPY_NDC                             ; Copy source to destination
```

```
                                012A   515          ;
                                012A   516          ;
                                012A   517          ;       For each logical-link to the remote node, copy the  ounters
                                012A   518          ;       and optionally update the "zero reference" set in  e LLI
                                012A   519          ;       using the formula:
                                012A   520          ;
                                012A   521          ;          SK     = SK + XWB_x + LLI_rt - LLI_lz
                                012A   522          ;          LLI_lz =      XWB_x + LLI_rt            ; iff zeroing
                                012A   523          ;
                                012A   524          ;
        5B  00000000'EF   D0   012A   525          MOVL     NET$GL_CNR_LLI,R11               ; Get logical-link CNR
                     5A   D4   0131   526          CLRL     R10                              ; Start at begining
                               0133   527 50$:     $SEARCH eql,lli,l,pna                     ; Find next LLI for remote node
                 37 50    E9   0142   528          BLBC     R0,100$                          ; If LBC, none
                               0145   529          ;
                               0145   530          ;
                               0145   531          ;          TEMP = XWB_x + LLI_rt
                               0145   532          ;
                               0145   533          ;
              52   2C AA  9E   0145   534          MOVAB    CNF+LLI$Z_NDC_RT(R10),R2         ; Point to LLI_rt
    51   24 AA  00000084 8F  C1   0149   535       ADDL3    #XWB$Z_NDC,CNF+LLI$L_XWB(R10),R1; Point to XWB_x
              50     02   D0   0152   536          MOVL     #COU_M_HIGHIPL,R0                ; Say "use hi IPL, don't zero"
                  FED4'   30   0155   537          BSBW     ADD_NDC_TEMP                     ; Return with TEMP in R2
              51     52   D0   0158   538          MOVL     R2,R1                            ; Copy TEMP to R1
                               015B   539          ;
                               015B   540          ;
                               015B   541          ;          SK = SK + TEMP
                               015B   542          ;
                               015B   543          ;
              52     56   D0   015B   544          MOVL     R6,R2                            ; Point to SK
                     50   D4   015E   545          CLRL     R0                               ; Say "don't zero or use hi IPL"
                  FEDD'   30   0160   546          BSBW     ADD_NDC                          ; Add TEMP to SK
                               0163   547          ;
                               0163   548          ;
                               0163   549          ;          SK = SK - LLI_lz
                               0163   550          ;
                               0163   551          ;
             51    48 AA  9E   0163   552          MOVAB    CNF+LLI$Z_NDC_LZ(R10),R1         ; Point to LLI_lz
                  FF53'   30   0167   553          BSBW     SUB_NDC                          ; Subtract from SK
                               016A   554          ;
                               016A   555          ;
                               016A   556          ;          LLI_lz  =  TEMP    (i.e., XWB_x + LLI_rt)
                               016A   557          ;
                               016A   558          ;
                 C6 54    E9   016A   559          BLBC     R4,50$                           ; If LBC, zeroing not requested
              52     51   D0   016D   560          MOVL     R1,R2                            ; Point to LLI_lz as destination
        51  00000000'EF   9E   0170   561          MOVAB    TEMP_Z_NDC,R1                    ; Point to TEMP as source
                  FE86'   30   0177   562          BSBW     COPY_NDC                         ; Copy TEMP to LLI_lz
                     B7   11   017A   563          BRB      50$                              ; Loop
                               017C   564                                                    ;
              50     01   D0   017C   565 100$:    MOVL     #1,R0                            ; Say "success"
              0E00 8F     BA   017F   566 200$:    POPR     #^M<R9,R10,R11>                  ; Restore regs
                     05        0183   567          RSB                                       ; Done
                               0184   568
```

```
                                 0184       .SBTTL  COPY_NDC                - Copy NDC counters
                                 0184  ;+
                                 0184  ;  This routine is called to copy the NDC counters from src to dest.
                                 0184  ;
                                 0184  ; Inputs:       R2      Non-pageable block to receive counters in NDC format
                                 0184  ;               R1      Source NDC counters
                                 0184  ;               R0      lbs => zero NDC counters
                                 0184  ;                       lbc => don't zero NDC counters
                                 0184  ;                       COU_V_HIGHIPL set => raise to NET$C_IPL
                                 0184  ;
                                 0184  ; Outputs:      All registers are preserved.
                                 0184  ;
                                 0184  ;-
                                 0184       .SAVE_PSECT
                             00000000       .PSECT  NET_LOCK_CODE,NOWRT,GBL
                                 0000
                                 0000  COPY_NDC:                                   ; NDC(R2)  <-  NDC(R1)
            0C 50    01    E1    0000       BBC     #COU_V_HIGHIPL,R0,70$          ; Br if no need to raise IPL
                                 0004       DSBINT  #NET$C_IPL                     ; Else, raise IPL now
                     04    10    000A       BSBB    70$                            ; Copy the counters
                                 000C       ENBINT                                 ; Restore IPL
                           05    000F       RSB                                    ; Return
                                 0010
                     3F    BB    0010  70$: PUSHR   #^M<R0,R1,R2,R3,R4,R5>         ; Save registers
         62    61    1C    28    0012       MOVC3   #NDC$C_LENGTH,(R1),(R2)        ; Move the counters
               50    6E    7D    0016       MOVQ    (SP),R0                        ; Get zero indicator, source area
               0D 50       E9    0019       BLBC    R0,90$                         ; Br if we don't zero the countes
      81    00000000'GF    D0    001C       MOVL    G^EXE$GL_ABSTIM,(R1)+          ; Else, reset the time last zeroed
61    18    00    6E    00    2C    0023       MOVC5   #0,(SP),#0,#NDC$C_LENGTH-4,(R1) ; And zero the counters
                     3F    BA    0029  90$: POPR    #^M<R0,R1,R2,R3,R4,R5>         ; Restore registers
                           05    002B       RSB                                    ; Done
                                 002C
```

```
                              002C            .SBTTL  ADD_NDC                 - Add NDC counters in NDC format
                              002C            .SBTTL  ADD_NDC_TEMP            - Add NDC counters & copy to temp area
                              002C      ;+
                              002C      ;  These routines are called to add NDC counters to a output buffer area.
                              002C      ;
                              002C      ; Inputs:       R2      Pointer to input counter area2
                              002C      ;                               Also the resultant storage area (ADD_NDC only)
                              002C      ;               R1      Pointer to input counter area1
                              002C      ;               R0      lbs => zero NDC counters ??? no-sense
                              002C      ;                       lbc => don't zero NDC counters
                              002C      ;                       COU_V_HIGHIPL set => raise to NET$C_IPL
                              002C      ;
                              002C      ; Outputs:      R2      Pointer to temp counter area (ADD_NDC_TEMP only)
                              002C      ;
                              002C      ;               All other registers are preserved.
                              002C      ;
                              002C      ;-
                              002C
                              002C      ADD_NDC_TEMP:                           ; temp <- NDC(R2)
                              002C                                              ; temp <- temp + NDC(R1)
                 51     DD    002C            PUSHL   R1                      ; Save counter area1
            51   52     D0    002E            MOVL    R2,R1                   ; Copy area2 pointer
   52    00000000'EF    9E    C 31            MOVAB   TEMP_Z_NDC,R2           ; Get temporary storage area address
                 C6     10    0038            BSBB    COPY_NDC                ; Get a copy of counter area2
                 51   8ED0    003A            POPL    R1                      ; Restore counter area1
                 01     10    003D            BSBB    ADD_NDC                 ; Get NDC(R2) <- NDC(R2) + NDC(R1)
                        05    003F            RSB                             ; Return
                              0040
                              0040
                              0040      ADD_NDC:                                ; NDC(R2)  <-  NDC(R1) + NDC(R2)
         0C 50   01     E1    0040            BBC     #COU_V_HIGHIPL,R0,70$   ; Br if no need to elevate IPL
                              0044            DSBINT  #NET$C_IPL              ; Else, raise IPL
                 04     10    004A            BSBB    70$                     ; Get sum of counters
                              004C            ENBINT                          ; Restore IPL
                        05    004F            RSB                             ; Return
                              0050
                              0050      70$:    ADDCOU  RSE,R1,R2,W             ; Add resource errors
                              005B            ADDCOU  RTO,R1,R2,W             ; Add response timeouts
                              0066            ADDCOU  CRC,R1,R2,W             ; Add connects received
                              0071            ADDCOU  CSN,R1,R2,W             ; Add connects sent
                              007C            ADDCOU  BRC,R1,R2,L             ; Add bytes received
                              0087            ADDCOU  BSN,R1,R2,L             ; Add bytes sent
                              0092            ADDCOU  PRC,R1,R2,L             ; Add packets received
                              009D            ADDCOU  PSN,R1,R2,L             ; Add packets sent
            11 50       E9    00A8            BLBC    R0,90$                  ; Br if zero not requested
                              00AB                                            ;
                 3F     BB    00AB            PUSHR   #^M<R0,R1,R2,R3,R4,R5>  ; Save regs
   81    00000000'GF    D0    00AD            MOVL    G^EXE$GL_ABSTIM,(R1)+   ; Else, reset the time last zeroed
61   18   00   6E  00   2C    00B4            MOVC5   #0,(SP),#0,#NDC$C_LENGTH-4,(R1) ; And zero the counters
                 3F     BA    00BA            POPR    #^M<R0,R1,R2,R3,R4,R5>  ; Restore regs
                              00BC                                            ;
                        05    00BC      90$:  RSB                             ; Done
                              00BD
```

```
                                  00BD          .SBTTL  SUB_NDC                 - Subtract NDC counters in NDC format
                                  00BD     ;+
                                  00BD     ;    This routine is called to subtract NDC counters from an output
                                  00BD     ;    buffer area.
                                  00BD     ;
                                  00BD     ; Inputs:         R2      Pointer to input counter area2
                                  00BD     ;                            Also the resultant storage area
                                  00BD     ;                 R1      Pointer to input counter area1
                                  00BD     ;                 R0      lbs => zero NDC counters ??? no-sense
                                  00BD     ;                         lbc => don't zero NDC counters
                                  00BD     ;                         COU_V_HIGHIPL set => raise to NET$C_IPL
                                  00BD     ;
                                  00BD     ; Outputs:        All registers are preserved.
                                  00BD     ;
                                  00BD     ;-
                                  00BD
                                  00BD     SUB_NDC:                                     ; NDC(R2)  <-  NDC(R2) - NDC(R1)
                 0C 50    01  E1  00BD          BBC     #COU_V_HIGHIPL,R0,70$          ; Br if no need to elevate IPL
                                  00C1          DSBINT  #NET$C_IPL                     ; Else, raise IPL
                          04  10  00C7          BSBB    70$                            ; Get sum of counters
                                  00C9          ENBINT                                 ; Restore IPL
                              05  00CC          RSB                                    ; Return
                                  00CD
                                  00CD     70$:  SUBCOU  RSE,R1,R2,W                   ; Subtract resource errors
                                  00D8          SUBCOU  RTO,R1,R2,W                    ; Subtract response timeouts
                                  00E3          SUBCOU  CRC,R1,R2,W                    ; Subtract connects received
                                  00EE          SUBCOU  CSN,R1,R2,W                    ; Subtract connects sent
                                  00F9          SUBCOU  BRC,R1,R2,L                    ; Subtract bytes received
                                  0104          SUBCOU  BSN,R1,R2,L                    ; Suutract bytes sent
                                  010F          SUBCOU  PRC,R1,R2,L                    ; Subtract packets received
                                  011A          SUBCOU  PSN,R1,R2,L                    ; Subtract packets sent
                       11 50  E9  0125          BLBC    R0,90$                         ; Br if zero not requested
                                  0128                                                 ;
                          3F  BB  0128          PUSHR   #^M<R0,R1,R2,R3,R4,R5>         ; Save regs
            81    00000000'GF  D0  012A          MOVL    G^EXE$GL_ABSTIM,(R1)+          ; Else, reset the time last zeroed
     61   18   00   6E    00  2C  0131          MOVC5   #0,(SP),#0,#NDC$C_LENGTH-4,(R1) ; And zero the counters
                          3F  BA  0137          POPR    #^M<R0,R1,R2,R3,R4,R5>         ; Restore regs
                                  0139                                                 ;
                              05  0139     90$:  RSB                                   ; Done
```

```
                                     013A          .SBTTL  LOG_NDCOU               - Log NDC counters
                                     013A     ;+
                                     013A     ;  This routine logs the NDC counters before re-using the NDC counter
                                     013A     ;  block.
                                     013A     ;
                                     013A     ; INPUTS:        R1      NDCOU block to be re-used
                                     013A     ;
                                     013A     ; OUTPUT:        R0      Garbage
                                     013A     ;
                                     013A     ;                All other registers are preserved.
                                     013A     ;-
                                     013A     LOG_NDCOU:
               0FFE 8F      BB       013A             PUSHR   #^M<R1,R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ; Save registers
          58     10 A1      3C       013E             MOVZWL  NDCOU$W_PNA(R1),R8        ; Get the remote node address
   5B    00000000'EF        D0       0142             MOVL    NET$GL_CNR_NDI,R11        ; Get root of NDI database
         00000000'EF        16       0149             JSB     NET$NDI_BY_ADD            ; Find CNF for old remote node
                 35 50      E9       014F             BLBC    R0,90$                    ; Exit on error
       51   019C 8F         3C       0152             MOVZWL  #LOGBUF_LEN+12,R1         ; Get length of buffer
                 FEA6'      30       0157             BSBW    NET$ALLOCATE              ; Allocate a buffer from ACP pool
                 2A 50      E9       015A             BLBC    R0,90$                    ; Exit on error
   00000000'FF      62      0E       015D             INSQUE  (R2),@NET$GQ_TMP_BUF      ; Insert buffer on tmp_buf queue.
          08 A2     51      B0       0164             MOVW    R1,NDCOU$W_SIZE(R2)       ; Set size of buffer
          53     0C A2      9E       0168             MOVAB   12(R2),R3                 ; Point to output buffer
                                     016C             $CNFFLD ndi,s,cnt,R9              ; Read counters request
                    06      A8       0173             BISW    #NET$M_CLRCNT!NET$M_LOGDBR,- ; Force counters to be logged
         00000000'EF                 0175                     NET$GL_FLAGS              ;         when read
         00000000'EF        16       017A             JSB     NET$NDI_S_CNT             ; Read & log the counters
                    06      AA       0180             BICW    #NET$M_CLRCNT!NET$M_LOGDBR,- ; Clean up flags
         00000000'EF                 0182                     NET$GL_FLAGS
               0FFE 8F      BA       0187     90$:    POPR    #^M<R1,R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ; Restore registers
                            05       018B             RSB                               ; Done
                                     018C
                                 00000184             .RESTORE_PSECT
                                     0184
```

```
                             0184            .SBTTL  GET_HASH_ADDR          - Get the table entry address
                             0184      ;+
                             0184      ;
                             0184      ;  INPUTS:       R3      Scratch
                             0184      ;                R2      Node address
                             0184      ;                R1      NDCOU address or zero if none
                             0184      ;                R0      Scratch
                             0184      ;
                             0184      ;  OUTPUTS:      R3      Address of NDCOU$L_LINK biased ptr to the NDCOU
                             0184      ;                R2      Hash Table address for entry
                             0184      ;                R1      Unchanged
                             0184      ;                R0      If LBS, NDCOU$L_LINK(R3) points to the R1 NDCOU
                             0184      ;                        If LBC, NDCOU$L_LINK(R3) contains a zero
                             0184      ;
                             0184      ;
                             0184      GET_HASH_ADDR:                                          ; Get the table entry address
                             0184              HASH_NODE_ADDRESS  R2                           ; Get Hash table address
                             019C                                                              ;
        50    F4 A2     9E   019C              MOVAB   -NDCOU$L_LINK(R2),R0                    ; Prepare for scan
           53    50     D0   01A0      10$:    MOVL    R0,R3                                   ; Update "previous NDCOU" ptr
        0C A3    51     D1   01A3              CMPL    R1,NDCOU$L_LINK(R3)                     ; Is this the previous NDCOU ?
                 0A     13   01A7              BEQL    90$                                     ; If EQL yes, we're done
        50    0C A3     D0   01A9              MOVL    NDCOU$L_LINK(R3),R0                     ; Get next entry
                 F1     12   01AD              BNEQ    10$                                     ; If NEQ, loop
                 50     D4   01AF              CLRL    R0                                      ; Say "R1 NDCOU not in list"
                 03     11   01B1              BRB     100$                                    ; Take common exit
           50    01     D0   01B3      90$:    MOVL    #1,R0                                   ; Say "R1 NDCOU was in list"
                        05   01B6      100$:   RSB                                             ; Done
                             01B7
                             01B7
                        01B7         .SBTTL  NET$LOOKUP_NDCOU        - Find NDCOU in Hash Table
                        01B7  ;+
                        01B7  ;
                        01B7  ;  INPUTS:        R8      Remote node address
                        01B7  ;                 R2      Scratch
                        01B7  ;                 R0      Scratch
                        01B7  ;
                        01B7  ;  OUPUTS:        R2      NDCOU pointer
                        01B7  ;                 R0      LBS if found
                        01B7  ;                         LBC otherwise
                        01B7  ;
                        01B7  ;                 All other registers are preserved.
                        01B7  ;
                        01B7  ;-
                        01B7  NET$LOOKUP_NDCOU:                       ; Find NDCOU in Hash Table
              50    D4  01B7          CLRL    R0                      ; Assume lookup failure
         52   58    D0  01B9          MOVL    R8,R2                   ; Get remote node address
                        01BC                                          ;
                        01BC          HASH_NODE_ADDRESS  R2           ; Get Hash Table address
                        01D4                                          ;
      52   F4 A2    9E  01D4          MOVAB   -NDCOU$L_LINK(R2),R2    ; Prepare for scan
      52   0C A2    D0  01D8  10$:    MOVL    NDCOU$L_LINK(R2),R2     ; Travel list
              08    13  01DC          BEQL    100$                    ; If EQL, at end of list
      10 A2   58    B1  01DE          CMPW    R8,NDCOU$W_PNA(R2)      ; Is this it ?
              F4    12  01E2          BNEQ    10$                     ; If NEQ no, loop
              50    D6  01E4          INCL    R0                      ; Say "NDCOU found"
                    05  01E6  100$:   RSB                             ; Done
                        01E7
                        01E7
                        01E7  .END
```

```
                                                             L 12
NETLLICNT                    - Counter support for nodes and logical- 16-SEP-1984 01:26:37  VAX/VMS Macro V04-00       Page  19
Symbol table                                                           5-SEP-1984 02:21:09  [NETACP.SRC]NETLLICNT.MAR;1      (15)

ACP$C_STA_F                   = 00000004                 LSB$V_LI                      = 00000000
ACP$C_STA_H                   = 00000005                 LSB$V_SPARE                   = 00000001
ACP$C_STA_I                   = 00000000                 LSB$W_HAA                     = 00000008
ACP$C_STA_N                   = 00000001                 LSB$W_HAR                     = 00000006
ACP$C_STA_R                   = 00000002                 LSB$W_HAX                     = 00000026
ACP$C_STA_S                   = 00000003                 LSB$W_HNR                     = 00000024
ADD_NDC                         00000040 R     05        LSB$W_HXS                     = 00000004
ADD_NDC_TEMP                    0000002C R     05        LSB$W_LNX                     = 00000002
BIT...                        = 00000008                 LSB$W_LUX                     = 00000000
BUG$_NETNOSTATE                 ********   X   04        NDC$C_LENGTH                  = 0000001C
CNF                           = 00000024                 NDC$L_ABS_TIM                 = 00000000
CNF$C_LENGTH                  = 00000024                 NDC$L_BRC                     = 0000000C
CNF$KEY_SEARCH                  ********   X   04        NDC$L_BSN                     = 00000010
CNF$_ADVANCE                  = 00000000                 NDC$L_PRC                     = 00000014
CNF$_QUIT                     = 00000002                 NDC$L_PSN                     = 00000018
CNF$_TAKE_CURR                = 00000003                 NDC$W_CRC                     = 00000008
CNF$_TAKE_PREV                = 00000001                 NDC$W_CSN                     = 0000000A
COPY_NDC                        00000000 R     05        NDC$W_RSE                     = 00000004
COU_M_HIGHIPL                 = 00000002                 NDC$W_RTO                     = 00000006
COU_M_ZERO                    = 00000001                 NDCOU$B_STS                     0000000B
COU_V_HIGHIPL                 = 00000001                 NDCOU$B_TYPE                    0000000A
COU_V_ZERO                    = 00000000                 NDCOU$C_LENGTH                = 00000030
EXE$GL_ABSTIM                   ********   X   04        NDCOU$L_LINK                    0000000C
GET_HASH_ADDR                   00000184 R     04        NDCOU$Q_LINKAGE                 00000000
HASH_C_LNG                    = 00000080                 NDCOU$W_PNA                     00000010
HASH_S_LNG                    = 00000007                 NDCOU$W_REFCNT                  00000012
HASH_V_LNG                    = 00000000                 NDCOU$W_SIZE                    00000008
LLI$C_XWB                     = 00000000                 NDCOU$Z_NDC                     00000014
LLI$Z_NDC_LZ                  = 00000024                 NDCOU_C_BLOCKS                = 00000200
LLI$Z_NDC_RT                  = 00000008                 NDCOU_C_SIZE                  = 00000030
LOGBUF_LEN                    = 00000190                 NET$ACQUIRE_NDCOU               0000002C RG    04
LOG_NDCOU                       0000013A R     05        NET$ALLOCATE                    ********   X   04
LSB                           = 00000000                 NET$C_ACT_TIMER               = 0000001E
LSB$B_R_CXBCNT                = 00000028                 NET$C_EFN_ASYN                = 00000002
LSB$B_R_CXBQUO                = 00000029                 NET$C_EFN_WAIT                = 00000001
LSB$B_SPARE                   = 0000002A                 NET$C_IPL                     = 00000008
LSB$B_STS                     = 0000002B                 NET$C_MAXACCFLD               = 00000027
LSB$B_X_ADJ                   = 0000000B                 NET$C_MAXLINNAM               = 0000000F
LSB$B_X_CXBACT                = 0000000D                 NET$C_MAXLNK                  = 000003FF
LSB$B_X_CXBCNT                = 0000000F                 NET$C_MAXNODNAM               = 00000006
LSB$B_X_CXBQUO                = 0000000E                 NET$C_MAXOBJNAM               = 0000000C
LSB$B_X_PKTWND                = 0000000C                 NET$C_MAX_AREAS               = 0000003F
LSB$B_X_REQ                   = 0000000A                 NET$C_MAX_LINES               = 00000040
LSB$L_CROSS                   = 0000002C                 NET$C_MAX_NCB                 = 0000006E
LSB$L_R_CXB                   = 00000020                 NET$C_MAX_NODES               = 000003FF
LSB$L_R_IRP                   = 0000001C                 NET$C_MAX_OBJ                 = 000000FF
LSB$L_X_CXB                   = 00000018                 NET$C_MAX_WQE                 = 00000014
LSB$L_X_IRP                   = 00000014                 NET$C_MINBUFSIZ               = 000000C0
LSB$L_X_PND                   = 00000010                 NET$C_TID_ACT                 = 00000003
LSB$M_BOM                     = 00000020                 NET$C_TID_RUS                 = 00000001
LSB$M_EOM                     = 00000040                 NET$C_TID_XRT                 = 00000002
LSB$M_LI                      = 00000001                 NET$C_TRCTL_CEL               = 00000002
LSB$S_LSB                     = 00000030                 NET$C_TRCTL_OVR               = 00000005
LSB$S_SPARE                   = 00000004                 NET$C_UTLBUFSIZ               = 00001000
LSB$S_STS                     = 00000001                 NET$FLUSH_LLI_CNT               000000D0 RG    04
LSB$V_BOM                     = 00000005                 NET$GL_CNR_LLI                  ********   X   04
LSB$V_EOM                     = 00000006                 NET$GL_CNR_NDI                  ********   X   05
```

```
NET$GL_FLAGS                ********   X    05         XWB$C_STA_CAR                  = 00000002
NET$GQ_TMP_BUF              ********   X    05         XWB$C_STA_CCS                  = 00000004
NET$GZ_HASHT_NDCOU          00000010 RG     03         XWB$C_STA_LIR                  = 00000003
NET$INIT_NDCOU              00000000 RG     04         XWB$C_STA_CIS                  = 00000001
NET$LOOKUP_NDCOU            000001B7 R      04         XWB$C_STA_CLO                  = 00000000
NET$M_CLRCNT              = 00000004                   XWB$C_STA_DIR                  = 00000006
NET$M_LOGDBR              = 00000002                   XWB$C_STA_DIS                  = 00000007
NET$M_MAXLNKMSK           = 000003FF                   XWB$C_STA_RUN                  = 00000005
NET$NDI_BY_ADD              ********   X    05         XWB$L_DEA_IRP                  = 00000104
NET$NDI_S_CNT               ********   X    05         XWB$L_FPC                      = 00000020
NET$Q_NDCOU_IDLE            00000000 R      03         XWB$L_FR3                      = 00000024
NET$Q_NDCOU_INACT           00000008 R      03         XWB$L_FR4                      = 00000028
NET$READ_LLI_CNT            000000F1 RG     04         XWB$L_ICB                      = 0000010C
NET$READ_NDI_CNT            00000113 RG     04         XWB$L_IRP_ACC                  = 00000080
NET$RELEASE_NDCOU           000000A6 RG     04         XWB$L_LINK                     = 0000002C
NFB$C_LLI_PNA             = 08010014                   XWB$L_ORGUCB                   = 00000010
NFB$C_NDI_CNT             = 02020042                   XWB$L_PID                      = 00000034
NFB$C_OP_EQL              = 00000000                   XWB$L_VCB                      = 00000030
NSP$C_EXT_LNK             = 0000001E                   XWB$L_WLBL                     = 00000004
NSP$C_MAXHDR              = 00000009                   XWB$L_WLFL                     = 00000000
PR$_IPL                     ********   X    05         XWB$M_FLG_BREAK                = 00000001
SIZ...                    = 00000006                   XWB$M_FLG_CLO                  = 00000200
SS$_INSFMEM                 ********   X    04         XWB$M_FLG_IAVL                 = 00001000
SUB_NDC                     000000BD R      05         XWB$M_FLG_SCD                  = 00000100
TEMP_Z_NDC                  00000000 R      02         XWB$M_FLG_SDACK                = 00000008
TR$C_MAXHDR               = 0000001C                   XWB$M_FLG_SDFL                 = 00004000
TR$C_NI_ALLEND1           = 040000AB                   XWB$M_FLG_SDT                  = 00000080
TR$C_NI_ALLEND2           = 00000000                   XWB$M_FLG_SIACK                = 00000004
TR$C_NI_ALLROU1           = 030000AB                   XWB$M_FLG_SIFL                 = 00002000
TR$C_NI_ALLROU2           = 00000000                   XWB$M_FLG_SLI                  = 00000010
TR$C_NI_PREFIX            = 000400AA                   XWB$M_FLG_TBPR                 = 00000800
TR$C_NI_PROT              = 00000360                   XWB$M_FLG_WBP                  = 00000040
TR$C_PRI_ECL              = 0000001F                   XWB$M_FLG_WBUF                 = 00000002
TR$C_PRI_RTHRU            = 0000001F                   XWB$M_FLG_WDAT                 = 00000400
XWB                       = 00000000                   XWB$M_FLG_WHGL                 = 00000020
XWB$B_ACCESS              = 0000000B                   XWB$M_PRO_CCA                  = 00000008
XWB$B_DATA                = 0000005B                   XWB$M_PRO_NAR                  = 00000010
XWB$B_FIPL                = 0000001F                   XWB$M_PRO_NFC                  = 00000001
XWB$B_LOGIN               = 000000CC                   XWB$M_PRO_PH2                  = 00000004
XWB$B_LPRNAM              = 000000A4                   XWB$M_PRO_SFC                  = 00000002
XWB$B_PRO                 = 0000005A                   XWB$M_STS_ASTPND               = 00000400
XWB$B_RID                 = 0000006F                   XWB$M_STS_ASTREQ               = 00000800
XWB$B_RPRNAM              = 000000B8                   XWB$M_STS_CON                  = 00000010
XWB$B_SP3                 = 0000006E                   XWB$M_STS_DIS                  = 00000008
XWB$B_STA                 = 0000001E                   XWB$M_STS_DTNAK                = 00000100
XWB$B_TYPE                = 0000000A                   XWB$M_STS_LINAK                = 00000200
XWB$B_X_FLW               = 0000006C                   XWB$M_STS_NDC                  = 00001000
XWB$B_X_FLWCNT            = 0000006D                   XWB$M_STS_OVF                  = 00000080
XWB$C_COMLNG              = 000000A4                   XWB$M_STS_RBP                  = 00000040
XWB$C_CONLNG              = 00000112                   XWB$M_STS_SOL                  = 00000004
XWB$C_DATA                = 00000010                   XWB$M_STS_TID                  = 00000001
XWB$C_LOGIN               = 00000040                   XWB$M_STS_TLI                  = 00000002
XWB$C_LPRNAM              = 00000014                   XWB$M_STS_TMO                  = 00000020
XWB$C_NDC_LNG             = 00000020                   XWB$Q_FORK                     = 00000014
XWB$C_NUMSTA              = 00000008                   XWB$Q_FREE_CXB                 = 00000118
XWB$C_RID                 = 00000010                   XWB$R_CON_BLK                  = 000000A4
XWB$C_RPRNAM              = 00000014                   XWB$R_RUN_BLK                  = 000000A4
```

```
XWB$S_                          = 00000006                  XWB$V_STS_TLI                   = 00000001
XWB$S_COMLNG                    = 0000006E                  XWB$V_STS_TMO                   = 00000005
XWB$S_CON_BLK                   = 0000006E                  XWB$W_CI_PATH                   = 00000110
XWB$S_DATA                      = 00000010                  XWB$W_DECAY                     = 0000004E
XWB$S_DT                        = 00000030                  XWB$W_DLY_FACT                  = 00000056
XWB$S_FLG                       = 00000002                  XWB$W_DLY_WGHT                  = 00000058
XWB$S_FORK                      = 00000008                  XWB$W_ELAPSE                    = 0000004A
XWB$S_FREE_CXB                  = 00000008                  XWB$W_FLG                       = 0000001C
XWB$S_LI                        = 00000030                  XWB$W_LOCLNK                    = 0000003E
XWB$S_LOGIN                     = 0000003F                  XWB$W_LOCSIZ                    = 00000040
XWB$S_LPRNAM                    = 00000013                  XWB$W_PATH                      = 00000038
XWB$S_NDC                       = 00000020                  XWB$W_PROGRESS                  = 00000052
XWB$S_PRO                       = 00000001                  XWB$W_REFCNT                    = 0000000C
XWB$S_RID                       = 00000010                  XWB$W_REMLNK                    = 0000003C
XWB$S_RPRNAM                    = 00000013                  XWB$W_REMNOD                    = 0000003A
XWB$S_RUN_BLK                   = 00000064                  XWB$W_REMSIZ                    = 00000042
XWB$S_STS                       = 00000002                  XWB$W_RETRAN                    = 00000054
XWB$S_XWB                       = 00000120                  XWB$W_R_REASON                  = 00000044
XWB$T_                          = 00000112                  XWB$W_SIZE                      = 00000008
XWB$T_DATA                      = 0000005C                  XWB$W_STS                       = 0000000E
XWB$T_DT                        = 000000A4                  XWB$W_TIMER                     = 00000050
XWB$T_LI                        = 000000D4                  XWB$W_TIM_ID                    = 00000048
XWB$T_LOGIN                     = 000000CD                  XWB$W_TIM_INACT                 = 0000004C
XWB$T_LPRNAM                    = 000000A5                  XWB$W_X_REASON                  = 00000046
XWB$T_RID                       = 00000070                  XWB$Z_NDC                       = 00000084
XWB$T_RPRNAM                    = 000000B9                  _$$_                            = 000000EF
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
```

```
                                                    +----------------+
                                                    ! Psect synopsis !
                                                    +----------------+

PSECT name                      Allocation          PSECT No.  Attributes
----------                      ----------          ---------  ----------
.  ABS  .                       00000000  (    0.)  00 (  0.)  NOPIC   USR   CON   ABS   LCL NOSHR NOEXE NORD  NOWRT NOVEC BYTE
$ABS$                           00000030  (   48.)  01 (  1.)  NOPIC   USR   CON   ABS   LCL NOSHR   EXE   RD    WRT NOVEC BYTE
NET_LOCK_IMPURE                 00000030  (   48.)  02 (  2.)  NOPIC   USR   CON   REL   LCL NOSHR NOEXE   RD    WRT NOVEC LONG
NET_IMPURE                      00000210  (  528.)  03 (  3.)  NOPIC   USR   CON   REL   LCL NOSHR NOEXE   RD    WRT NOVEC QUAD
NET_CODE                        000001E7  (  487.)  04 (  4.)  NOPIC   USR   CON   REL   LCL NOSHR   EXE   RD  NOWRT NOVEC BYTE
NET_LOCK_CODE                   0000018C  (  396.)  05 (  5.)  NOPIC   USR   CON   REL   GBL NOSHR   EXE   RD  NOWRT NOVEC BYTE

                                                 +------------------------+
                                                 ! Performance indicators !
                                                 +------------------------+

Phase                        Page faults    CPU Time       Elapsed Time
-----                        -----------    --------       ------------
Initialization                        29    00:00:00.07    00:00:00.39
Command processing                   128    00:00:01.02    00:00:05.20
Pass 1                               367    00:00:11.61    00:00:18.34
Symbol table sort                      0    00:00:01.49    00:00:01.58
Pass 2                               145    00:00:02.71    00:00:03.41
Symbol table output                   37    00:00:00.25    00:00:00.43
Psect synopsis output                  3    00:00:00.03    00:00:00.03
Cross-reference output                 0    00:00:00.00    00:00:00.00
Assembler run totals                 711    00:00:17.19    00:00:29.40

The working set limit was 1950 pages.
62033 bytes (122 pages) of virtual memory were used to buffer the intermediate code.
There were 60 pages of symbol table space allocated to hold 1086 non-local and 42 local symbols.
792 source lines were read in Pass 1, producing 21 object records in Pass 2.
38 pages of virtual memory were used to define 28 macros.

                                                +--------------------------+
                                                ! Macro library statistics !
                                                +--------------------------+

Macro library name                               Macros defined
------------------                               --------------
_$255$DUA28:[SHRLIB]NMALIBRY.MLB;1                          0
_$255$DUA28:[SHRLIB]EVCDEF.MLB;1                            1
_$255$DUA28:[NETACP.OBJ]NETDRV.MLB;1                        1
_$255$DUA28:[NETACP.OBJ]NET.MLB;1                           8
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                              3
_$255$DUA28:[SYSLIB]STARLET.MLB;2                           7
TOTALS (all libraries)                                     20

1238 GETS were required to define 20 macros.

There were no errors, warnings or information messages.

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

0278 AH-BT13A-SE DIGITAL EQUIPMENT CORPORATION
VAX/VMS V4.0 CONFIDENTIAL AND PROPRIETARY