

NNN		NNN	CCCCCCCCCCCC	PPPPPPPPPPPP	
NNN		NNN	CCCCCCCCCCCC	PPPPPPPPPPPP	
NNN		NNN	CCCCCCCCCCCC	PPPPPPPPPPPP	
NNN		NNN	CCC	PPP	PPP
NNN		NNN	CCC	PPP	PPP
NNN		NNN	CCC	PPP	PPP
NNNNNN		NNN	CCC	PPP	PPP
NNNNNN		NNN	CCC	PPP	PPP
NNNNNN		NNN	CCC	PPP	PPP
NNN	NNN	NNN	CCC	PPPPPPPPPPPP	
NNN	NNN	NNN	CCC	PPPPPPPPPPPP	
NNN	NNN	NNN	CCC	PPPPPPPPPPPP	
NNN	NNNNNN	NNN	CCC	PPP	
NNN	NNNNNN	NNN	CCC	PPP	
NNN	NNNNNN	NNN	CCC	PPP	
NNN	NNN	NNN	CCC	PPP	
NNN	NNN	NNN	CCC	PPP	
NNN	NNN	NNN	CCC	PPP	
NNN	NNN	NNN	CCCCCCCCCCCC	PPP	
NNN	NNN	NNN	CCCCCCCCCCCC	PPP	
NNN	NNN	NNN	CCCCCCCCCCCC	PPP	

```

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
LLLLLLLLLLLLLL        IIIIII                      SSSSSSSS
LLLLLLLLLLLLLL        IIIIII                      SSSSSSSS

```


(1)	63	DECLARATIONS
(1)	128	READ ONLY DATA
(1)	211	READ/WRITE DATA
(1)	331	ACT\$VRB_CONNECT INIT REMOTE CONSOLE DIALOGUE

```
0000 1 .TITLE NCPCONCAR - Console Carrier Requester Ethernet I/O
0000 2 .IDENT 'V04-000'
0000 3
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 :++
0000 29 : FACILITY: REMOTE CONSOLE CARRIER (NI)
0000 30 :
0000 31 : ABSTRACT:
0000 32 :
0000 33 : THIS PROGRAM RUNS ON A LOCAL NODE TO ALLOW A TERMINAL TO APPEAR TO
0000 34 : BE LOCALLY CONNECTED TO A REMOTE CONSOLE.
0000 35 :
0000 36 :
0000 37 : ENVIRONMENT: VMS - USER MODE
0000 38 :
0000 39 :
0000 40 : AUTHOR: Scott G. Davis, 17-Mar-1983
0000 41 :
0000 42 : MODIFIED BY:
0000 43 :
0000 44 : V004 PRD0110 Paul R. DeStefano 30-Jul-1984
0000 45 : Store SERVICE PASSWORD in NICE message in image
0000 46 : (HI-n) format.
0000 47 :
0000 48 : V003 TMH0003 Tim Halvorsen 28-Jul-1984
0000 49 : Deassign channels to XEDRIVER before issuing request
0000 50 : to down-line load console carrier microcode. This is
0000 51 : necessary because NDDRIVER can't operate on a XE UCB
0000 52 : which isn't fully owned by NETACP.
0000 53 :
0000 54 : V002 SGD0002 Scott G. Davis 07-Mar-1984
0000 55 : Send SERVICE PASSWORD in Reserve Console message.
0000 56 :
0000 57 : V001 SGD0001 Scott G. Davis 30-Aug-1983
```


NCPCONCAR
V04-000

- Console Carrier Requester Ethernet I/O 16-SEP-1984 01:58:40 VAX/VMS Macro V04-00
5-SEP-1984 02:13:29 [NCP.SRC]NCPCONCAR.MAR;1

Page 2
(1)

0000 58 :
0000 59 :
0000 60 :**
0000 61 :--

Correctly store SERVICE PASSWORD in NICE message.

```
0000 63 .SBTTL DECLARATIONS
0000 64 :
0000 65 : DEFAULT ADDRESSING MODE
0000 66 :
0000 67 : .DEFAULT DISPLACEMENT WORD
0000 68 :
0000 69 : INCLUDE FILES:
0000 70 :
0000 71 $NMADEF ; Network management definitions
0000 72 $NFBDEF ; Network function block definitions
0000 73 :
0000 74 : MACROS:
0000 75 :
0000 76 .MACRO ONERROR ACTION,?L
0000 77 BLBS RO,L
0000 78 ACTION
0000 79 L:
0000 80 .ENDM ONERROR
0000 81 :
0000 82 : EQUATED SYMBOLS:
0000 83 :
0000 84 :
0000 85 : ***NOTE References to PDBs assume they are of the form:
0000 86 :
0000 87 : .BYTE presence-flag
0000 88 : .BLKB n - data
0000 89 :
0000 90 :
00000004 0000 91 DONE_CHAR = 04 ; Character to finish I/O (^D)
00000002 0000 92 BREAK_CHAR = 02 ; Character to set "break" (^B)
0000000A 0000 93 LF = 10
0000000D 0000 94 CR = 13
00000005 0000 95 RETRY_CNT = 5 ; No. of times to retry connect
00000005 0000 96 TIMER = 5 ; No. of seconds between retries
0000 97 :
0000001C 0000 98 NFB$SIZ = NFB$C_LENGTH+12 ; Network function block size
00000056 0000 99 P2BUFSIZ = NFB$C_CTX_SIZE+4+18 ; P2 minimum + service circuit string
00000200 0000 100 P4BUFSIZ = 512
00000200 0000 101 XMTBUFLEN = 512 ; Size of transmit buffer
00000200 0000 102 RCVBUFLEN = 512 ; Size of receive buffer
0000 103 :
00000005 0000 104 MOP$C_REQID = 5 ; Request ID
00000007 0000 105 MOP$C_SYSTEMID = 7 ; System ID
0000000D 0000 106 MOP$C_RESERVE = 13 ; Reserve console
0000000F 0000 107 MOP$C_RELEASE = 15 ; Release console
00000011 0000 108 MOP$C_COMMAND = 17 ; Command and poll
00000001 0000 109 MOP$M_MSGNUM = 1 ; Message number bit
00000002 0000 110 MOP$M_BREAK = 2 ; Break indicator bit
00000013 0000 111 MOP$C_RESPONSE = 19 ; Response and acknowledge
00000001 0000 112 MOP$V_CMDLOST = 1 ; Message lost bit
0000 113 :
00000002 0000 114 MOP$C_INFO_FUNC = 2 ; INFO type code - functions
00000005 0000 115 MOP$V_FUNC_CARR = 5 ; Console carrier active
00000007 0000 116 MOP$V_FUNC_RSRV = 7 ; Console reserved
00000002 0000 117 MOP$C_FUNC_SIZE = 2 ; Size of field
0000 118 :
00000003 0000 119 MOP$C_INFO_USER = 3 ; INFO type code - console user
```


NCPCONCAR
V04-000

- Console Carrier Requester Ethernet I/O 16-SEP-1984 01:58:40 VAX/VMS Macro V04-00 Page 4
DECLARATIONS 5-SEP-1984 02:13:29 [NCP.SRC]NCPCONCAR.MAR;1 (1)

00000004	0000	120	MOPSC_INFO_TIMR =	4	; INFO type code - reservation timer
00000005	0000	121	MOPSC_INFO_CSIZ =	5	; INFO type code - console command size
00000006	0000	122	MOPSC_INFO_RSIZ =	6	; INFO type code - console response size

```
0000 124 :  
0000 125 : OWN STORAGE:  
0000 126 :  
00000000 127 : .PSECT CONCAR$PURE,NOWRT,NOEXE  
0000 128 : .SBTTL READ ONLY DATA  
0000 129 :  
0000 130 : ASSUME BREAK_CHAR LT 32  
0000 131 : ASSUME DONE_CHAR LT 32  
0000 132 :  
00000000 0000 133 TERMINATOR_DESC: .LONG 0 ; For trapping terminators  
00000014 0004 134 .LONG <1@BREAK_CHAR>!<1@DONE_CHAR>  
0000001C 0008 135  
0000056F 000C 136 CON$NFB_DESC:: .LONG NFBSIZ ; NFB descriptor  
0010 137 .ADDRESS CON$NFB  
00000056 0010 138  
0000058B 0014 139 CON$P2BUF_DESC:: .LONG P2BUFSIZ ; P2BUF descriptor  
0018 140 .ADDRESS CON$P2BUF  
00000200 0018 141  
000005E1 001C 142 CON$P4BUF_DESC:: .LONG P4BUFSIZ ; P4BUF descriptor  
0020 143 .ADDRESS CON$P4BUF  
0020 144  
0000003C 0020 145 SETPARAM_DESC:  
000004BE 0024 146 .LONG SETPARAMLEN  
0028 147 .ADDRESS SETPARAM  
0028 148  
FFFFFFFF FD050F80 0028 149 TIMEOUT: .LONG -10*1000*1000*TIMER,-1 ; Timer for NI responses during setup  
0030 150  
0030 151 :*****  
0030 152 : Note:  
0030 153 : The arrays NICE_PARAM_ADDR/ID have parallel entries.  
0030 154 :*****  
000007E3 0030 155 NICE_PARAM_ADDR: ; Addresses to store param values  
00000000 0034 156 .ADDRESS ADDR_HARD ; Target Ethernet hardware address  
00000000 0038 157 .ADDRESS PDB$G_CON_SLI ; NI device to use  
00000000 003C 158 .ADDRESS PDB$G_CON_SPW ; For RESERVE CONSOLE and LOAD messages  
0040 159 .LONG 0 ; End of table  
0040 160  
0072 0040 161 NICE_PARAM_ID: ; Table of parameters in charac response  
006E 0042 162 .WORD NMASC_PCNO_HWA ; Hardware address  
006F 0044 163 .WORD NMASC_PCNO_SLI ; Service circuit  
0046 164 .WORD NMASC_PCNO_SPA ; Service password  
0046 165  
05 0046 166 REQUEST_ID_MSG: ; To poll for presence of console  
00 0047 167 .BYTE MOP$C_REQID ; Code  
00 0048 168 .BYTE 0 ; Reserved byte, for future use  
00000004 004A 169 .BYTE 0,0 ; Receipt number - **What goes here?**  
004A 170 REQUEST_MSG_LEN = .-REQUEST_ID_MSG  
004A 171  
0F 004A 172 RELEASE_MSG: ; To release console  
00000001 004B 173 .BYTE MOP$C_RELEASE ; Code  
004B 174 RELEASE_MSG_LEN = .-RELEASE_MSG  
004B 175  
00000031 004B 176 CONNECTED_MSG: .LONG 20$-10$ ; Prompt  
00000053 004F 177 .LONG 10$  
6E 6F 63 20 65 6C 6F 73 6E 6F 43 0A 0053 178 10$: .ASCII <LF>\Console connected (press CTRL/D when finished)\<CR><LF>  
73 65 72 70 28 20 64 65 74 63 65 6E 005F  
65 68 77 20 44 2F 4C 52 54 43 20 73 006B
```



```
0D 29 64 65 68 73 69 6E 69 66 20 6E 0077
OA 0083
0084 179 20$:
0084 180
00000002' 0084 181 FINISH_MSG: .LONG 20$-10$ ; Message when done
0000008C' 0088 182 .ADDRESS 10$
OA 0D 008C 183 10$: .ASCII <CR><LF>
008E 184 20$:
008E 185
0000001A' 008E 186 NO_RESP_MSG: .LONG 20$-10$ ; Status message
00000096' 0092 187 .LONG 10$
73 65 6F 64 20 74 65 67 72 61 54 0A 0096 188 10$: .ASCII <LF>/Target does not respond/<CR><LF>
64 6E 6F 70 73 65 72 20 74 6F 6E 20 00A2
OA 0D 00AE
00B0 189 20$:
00B0 190
0000001C' 00B0 191 NO_HARD_MSG: .LONG 20$-10$ ; Hardware address required message
000000B8' 00B4 192 .LONG 10$
64 61 20 65 72 61 77 64 72 61 48 0A 00B8 193 10$: .ASCII <LF>/Hardware address required/<CR><LF>
72 69 75 71 65 72 20 73 73 65 72 64 00C4
OA 0D 64 65 00D0
00D4 194 20$:
00D4 195
00000011' 00D4 196 IN_USE_MSG: .LONG 20$-10$ ; Status message
000000DC' 00D8 197 .LONG 10$
20 6E 69 20 65 6C 6F 73 6E 6F 43 0A 00DC 198 10$: .ASCII <LF>/Console in use/<CR><LF>
OA 0D 65 73 75 00E8
00ED 199 20$:
00ED 200
4F 43 24 53 59 53 000000F5'010E0000' 00ED 201 TTY_DESC: .ASCID /SYS$COMMAND/
44 4E 41 4D 4D 00FB
0100 202
30 30 33 0100 203 NETMANVER: .ASCII /300/ ; Network management version 3.0.0
0103 204
43 43 4F 54 55 4C 50 00' 0103 205 LOA_FILE: .ASCIC /PLUTOCC/ ; Name of console carrier file
07 0103
00000008 010B 206 LOA_LTH = .-LOA_FILE
010B 207
4C 57 4F 54 55 4C 50 00' 010B 208 SLO_FILE: .ASCIC /PLUTOWL/ ; Name of secondary loader
07 010B
00000008 0113 209 SLO_LTH = .-SLO_FILE
```

```
0113 211 .SBTTL READ/WRITE DATA
00000000 212 .PSECT CONCAR$IMPURE,NOEXE,WRT,LONG
00000008 0000 213
00000010 0008 214 TTY_IN_IOSB: .BLKQ 1 ; Terminal input status block
00000018 0010 215 IOSB: .BLKQ 1 ; I/O status block
00000018 0010 216
00000018 0018 217 LOAD_RESP_DESC: .BLKQ 1 ; Points to NML response to load request
0000009C 0018 218
0000001C 009C 219 SYSTEM_ID_MSG: .BLKB 128+4 ; Enough room for buffer + CRC
000000B4 009C 220 SYSTEM_INFO = .SYSTEM_ID_MSG+4 ; Start of INFO section of message
000000A0 009C 221 SYSTEM_ID_LTH = .-SYSTEM_ID_MSG
000000A0 009C 222 SYSTEM_ID_END: .BLKL 1 ; Holds pointer to end of SYSTEMID msg
000002A4 00A0 223
000002A4 02A4 224 NI_RCVBUF: .BLKB RCVBUFLen+4 ; Enough room for buffer + CRC
000002A6 02A4 225
000002A6 02A5 226 NI_XMTBUF: .BYTE MOP$C_COMMAND ; This is a COMMAND/POLL message
00000002 02A6 227 MOP_CTRL_FLAGS: .BLKB 1 ; Flags byte: bit 0 = msg no.
000002A6 02A6 228 ; bit 1 = break flag
000004A6 02A6 229 COMMAND_OHD = .-NI_XMTBUF ; Amount of overhead in message
000004A6 02A6 230 TTY_INBUF = NI_XMTBUF+COMMAND_OHD ; Terminal input comes here
000004A6 04A6 231 .BLKB XMTBUFLen ; Start of xmit data
000004AA 04A6 232
000004AA 04AA 233 CON$XE_DESCADDR: .BLKA 1 ; Address of NI device descriptor
000004AE 04AA 234
000004AE 04AE 235 RETRY_COUNTER: .BLKL 1 ; For making sure of responses
000007E5' 04AE 236
000004FB' 04B2 237 NI_ADDRESSES: .ADDRESS ADDR_HARD+2 ; Hardware address
000004B8 04B6 238 .ADDRESS ADDR_PHYS+1 ; Physical address
000004BA 04B8 239 NICHAN_HARD: .BLKW 1 ; NI channel for hardware address
000004BA 04B8 240 NICHAN_PHYS: .BLKW 1 ; NI channel for physical address
000004BC 04BA 241 ; *** NICHAN HARD/PHYS must be together in this order, for context indexing
000004BC 04BC 242 NICHAN: .BLKW 1 ; For I/O operations
000004BE 04BC 243
000004BE 04BE 244 TTYCHAN: .BLKW 1 ; Terminal I/O channel
000004BE 04BE 245
000004BE 04BE 246 SETPARAM:
00000200 04C0 247 .WORD NMASC_PCLI_BUS ; Buffer size
00000451 04C4 248 .LONG RCVBUFLen
00000007 04C6 249 .WORD NMASC_PCLI_BFN ; Number of buffers
00000000 04CA 250 .LONG 7
00000000 04CC 251 .WORD NMASC_PCLI_PAD ; Pad short buffers
00000000 04D0 252 .LONG NMASC_STATE_ON
00000260 04D2 253 .WORD NMASC_PCLI_PTY ; Protocol type
00000018 04D6 254 .LONG ^X0260 ; Console carrier protocol type
00000001 04D8 255 .WORD NMASC_PCLI_PRM ; Promiscuous mode disabled
00000001 04DC 256 .LONG NMASC_STATE_OFF
00000001 04DE 257 .WORD NMASC_PCLI_DCH ; Data chaining off
00000001 04E2 258 .LONG NMASC_STATE_OFF
00000000 04E4 259 .WORD NMASC_PCLI_CRC ; Generate crc on transmit
00000000 04E8 260 .LONG NMASC_STATE_ON
00000002 04EA 261 .WORD NMASC_PCLI_ACC ; Protocol access mode
00000002 04EE 262 .LONG NMASC_ACC_LIM ; Limited sharing
00000008 04F0 263 .WORD NMASC_PCLI_DES ; Destination addr for limited sharing
00000001 04F2 264 .WORD 8 ; Count for parameter
000004FA 04F4 265 .WORD NMASC_LINMC_SET ; Set the address that follows
000004FA 04FA 266 DEST_ADDR: .BLKB 6 ; Space for complete NI ADDRESS
000004FA 04FA 267 ; *** NOTE:
```



```
04FA 268 : 1. DEST_ADDR must be right after DES
04FA 269 : 2. DEST_ADDR is multiplexed. It is used initially to specify the destination
04FA 270 : address for protocol sharing. Thereafter, it contains the pointer to
04FA 271 : address to be used for Ethernet transmits.
04FA 272 : ***
0000003C 04FA 273 SETPARAMLEN=-SETPARAM ; **line to be deleted later
04FA 274
06 04FA 275 ADDR_PHYS: .BYTE 6 ; Counted physical Ethernet address
AA 04FB 276 .BYTE ^XAA ; Four wired bytes for DNA addresses
00 04FC 277 .BYTE 00
04 04FD 278 .BYTE 04
00 04FE 279 .BYTE 00
00000501 04FF 280 CON$W_ADDR_DNA:: .BLKW 1 ; For storing DNA node address, in
0501 281 ; reverse byte order
0501 282 RESERVE_MSG: ; To reserve console
0501 283 .BYTE MOP$C_RESERVE ; Code
0000050A 0502 284 .BLKB 8 ; Room for service password
00000009 050A 285 RESERVE_MSG_LEN = .-RESERVE_MSG
050A 286
3A 30 41 45 58 00000512'010E0000' 050A 287 XE_DESC: .ASCID /XEA0:/ ; Descriptor of service circuit
0517 288 ; XEA0: is the default
0000051B 0517 289 NICE_SHOW_DESC: .BLKL 1 ; Length of NICE SHOW message
0000051F' 051B 290 .ADDRESS NICE_SHOW_MSG
14 051F 291 NICE_SHOW_MSG: .BYTE NMA$C_FNC_REA ; Read function
20 0520 292 .BYTE NMA$C_OPINF CHA04!NMA$C_ENT_NOD ; Get node characteristics
00000528 0521 293 CONST_SHOW_NODE:: .BLKB 7 ; Leave room for node name
0528 294
0000052C 0528 295 NICE_LOAD_DESC: .BLKL 1 ; Length of load message
00000530' 052C 296 .ADDRESS NICE_LOAD_MSG ; Address of load message
0530 297 NICE_LOAD_MSG: ; Given to NMLSHR to load console code
16 0530 298 .BYTE NMA$C_FNC_SYS ; System-specific
04 0531 299 .BYTE NMA$C_SYS_VMS ; for VMS
OF 0532 300 .BYTE NMA$C_FNC_LOA ; Load function
0533 301 LOAD_PARAMS: ; For parameters
00000534 0533 302 .BLKB 1 ; Option - load by node/circuit
0000053D 0534 303 .BLKB 2+7 ; Space for physical address, maybe
00000550 053D 304 .BLKB 2+17 ; Space for service circuit - variable
0000055B 0550 305 .BLKB 2+9 ; Space for param ID+password
00000565 055B 306 .BLKB 2+LOA_LTH ; Space for param ID+load file
0000056F 0565 307 .BLKB 2+SLO_LTH ; Space for param ID+sec loader file
056F 308
056F 309 ZERO_START: ; Storage to be initialized
056F 310
0000058B 056F 311 CON$NFB:: .BLKB NFBSIZ ; Network function block
058B 312
000005E1 058B 313 CON$P2BUF:: .BLKB P2BUFSIZ ; Buffer for describing what I want
05E1 314
000007E1 05E1 315 CON$P4BUF:: .BLKB P4BUFSIZ ; Buffer for getting what I want
07E1 316
000007E2 07E1 317 NICE_MSG_FLAG: .BLKB 1 ; Error flag for processing NICE msg
07E2 318
000007E3 07E2 319 MOP_MSG_NUM: .BLKB 1 ; Message number -1 for command msgs
07E3 320
000007EB 07E3 321 ADDR_HARD: .BLKB 8 ; Hardware address of target. Format:
07EB 322 ; Byte 1 - existence flag
07EB 323 ; Byte 2 - "6" (length of string)
07EB 324 ; Byte 3-8 - Ethernet address
```

NCPCONCAR
V04-000

- Console Carrier Requester Ethernet I/O 16-SEP-1984 01:58:40 VAX/VMS Macro V04-00 Page 9
READ/WRITE DATA 5-SEP-1984 02:13:29 [NCP.SRC]NCPCONCAR.MAR;1 (1)

0000027C	07EB	325	ZERO_LTH	=	.-ZERO_START	
	07EB	326				
000007F9	07EB	327	MY_ADDRESS:	.BLKB	14	
	07F9	328				

; Place to hold Ethernet header, starting
; with my address


```
00000000 330 .PSECT CONCAR$CODE,NOWRT
0000 331 .SBTTL ACT$VRB_CONNECT INIT REMOTE CONSOLE DIALOGUE
0000 332 :++
0000 333 : FUNCTIONAL DESCRIPTION:
0000 334 :
0000 335 : Initiate dialogue with remote console:
0000 336 : - Attempt to reserve console
0000 337 : - Request System ID
0000 338 : - Determine whether you have it.
0000 339 :   - If so - proceed with exchanging characters
0000 340 :   - If not - request downline load of console code, etc.
0000 341 :
0000 342 : CALLING SEQUENCE:
0000 343 :
0000 344 : CALLS #N,ACT$VRB_CONNECT
0000 345 :
0000 346 : INPUT PARAMETERS:
0000 347 :
0000 348 : NONE
0000 349 :
0000 350 : IMPLICIT INPUTS:
0000 351 :
0000 352 : NONE
0000 353 :
0000 354 : OUTPUT PARAMETERS:
0000 355 :
0000 356 : NONE
0000 357 :
0000 358 : IMPLICIT OUTPUTS:
0000 359 :
0000 360 : NONE
0000 361 :
0000 362 : COMPLETION CODES:
0000 363 :
0000 364 : SIDE EFFECTS:
0000 365 :
0000 366 : Console carrier maintenance link set up with remote node
0000 367 :
0000 368 :--
0000 369 :
OFFC 0000 370 ACT$VRB_CONNECT:: .WORD ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
0002 371 $ASSIGN_S DEVNAM = TTY_DESC, CHAN = TTYCHAN ; Get terminal channel
0013 372 ONERROR_RET ; If error done
027C 8F 00 056F'CF 00 2C 0017 373 MOVCS #0,ZERO_START,#0,#ZERO_LTH,ZERO_START ; Init storage
0020 374 :
0023 375 : Get parameter values supplied by NCP command
0023 376 :
0023 377 PUSHAB NETMANVER ; Addr of management version string
00000000'0100'CF 9F 0023 378 CALLS #1,G^NML$INITIALIZE ; Initialize NMLSHR
0000'GF 01 FB 0027 379
002E 380 ASSUME NMASC_ENT_NOD EQ 0
002E 381
0000'CF D5 002E 382 TSTL NCP$GL_OPTION ; Is this CONNECT by node?
003F 12 0032 383 BNEQ 6$ ; If NEQ no - don't look anything up
0327'CF 9F 0034 384 PUSHAB CON$SHOW MSG ; Address of message processing routine
0517'CF 9F 0038 385 PUSHAB NICE_SHOW_DESC ; Address of descriptor of my READ
```

```
53 0521'CF 9E 003C 386 MOVAB CONST SHOW_NODE,R3 ; Point to storage for node-id
57 051F'CF 9E 0041 387 MOVAB NICE_SHOW_MSG,R7 ; Point to beginning of message
56 0001'CF 9E 0046 388 MOVAB PDB$G_VRB_ENT+1,R6 ; Point to node-id
58 86 9A 004B 389 MOVZBL (R6)+,R8 ; Was a node address specified?
07 12 004E 390 BNEQ 2$ ; if NEQ no - use node name
0050 391
0050 392 ASSUME NMASC_ENT_NOD EQ 0
0050 393
83 83 94 0050 394 CLRB (R3)+ ; Set parameter type
83 66 80 0052 395 MOVW (R6),(R3)+ ; Store node address
07 11 0055 396 BRB 4$ ; Continue in common code
0057 397
83 58 90 0057 398 2$: MOVB R8,(R3)+ ; Store length of name
63 66 58 28 005A 399 MOVCB R8,(R6),(R3) ; Store node name
0517'CF 53 57 C3 005E 400 4$: SUBL3 R7,R3,NICE_SHOW_DESC ; Store length of message
00000000'GF 02 FB 0064 401 CALLS #2,G^NML$PROCESS_NICE ; Ship the message
03 07E1'CF E9 006B 402 BLBC NICE_MSG_FLAG,6$ ; If LBC no error
020D 31 0070 403 BRW 2000$ ; Terminate processing
0073 404
0073 405 ; Do any necessary parameter conversion
0073 406
04A6'CF 050A'CF 9E 0073 407 6$: MOVAB XE_DESC,CON$XE_DESCADDR ; Default to XEA0: (UNA-0)
0000'CF 95 007A 408 TSTB PDB$G_CON_SLI ; Was service circuit specified?
03 13 007E 409 BEQL 10$ ; If EQL no
0080 410
0080 411 ; Convert SLI to XEq0:
FF7D' 30 0080 412
0083 413 BSBW CON$GET_SLI ; Do ultimate call to NETACP
0083 414
0083 415 ; Set up channels for request IDs
0083 416
0083 417
52 0002'CF 9E 0083 418 10$: MOVAB PDB$G_CON_PHA+2,R2 ; Point to physical address, maybe
FE A2 95 0088 419 TSTB -2(R2) ; Was a physical address provided?
28 12 008B 420 BNEQ 20$ ; If NEQ yes - use it alone
07E3'CF 95 008D 421 TSTB ADDR_HARD ; Is there a hardware address?
10 12 0091 422 BNEQ 15$ ; If NEQ yes - use it alone
53 00B4'CF D0 0093 423 MOVL NO_HARD_MSG+4,R3 ; Address of message
52 00B0'CF D0 0098 424 MOVL NO_HARD_MSG,R2 ; Length of message
0331 30 009D 425 BSBW NCP$TTY_WRITE ; Output the message
01C5 31 00A0 426 BRW 1500$ ; Done
52 07E5'CF 9E 00A3 427 15$: MOVAB ADDR_HARD+2,R2 ; Assume I have a hardware address
56 04B6'CF 9E 00A8 428 MOVAB NICHAN_HARD,R6 ; Address of channel to physical address
02C9 30 00AD 429 BSBW SETUP_NICHAN ; Set up the Ethernet channel
52 04FB'CF 9E 00B0 430 MOVAB ADDR_PHYS+1,R2 ; Point to physical address
56 04B8'CF 9E 00B5 431 20$: MOVAB NICHAN_PHYS,R6 ; Address of channel to hardware address
02BC 30 00BA 432 BSBW SETUP_NICHAN ; Set up the Ethernet channel
00BD 433
00BD 434 ; Try to get the console connected
00BD 435
04AA'CF 05 D0 00BD 436 MOVL #RETRY_CNT,RETRY_COUNTER ; Set up number of times to retry
00C2 437
00C2 438 ; *****
00C2 439 ; Need initial checking code:
00C2 440 1. Request system ID
00C2 441 2. If no response or console not loaded, done
00C2 442 3. If console owned by some else, too bad
00C2 443 4. If console owned by me, bug: release console
```



```
00C2 443 :*****
00C2 444 :
00C2 445 : Transmit RESERVE message, to see if I can get console
00C2 446 :
04BA'CF 56 01 D0 00C2 447 100$: MOVL #1,R6 ; Set to flipflop channels
04B6'CF 46 B0 00C5 448 150$: MOVW NICHAN_HARD[R6],NICHAN ; Use the proper channel
4C 13 00CD 449 : BEQL 410$ ; If EQL, channel was never assigned
04F4'CF 04AE'CF 46 D0 00CF 450 : MOVL NI_ADDRESSES[R6],DEST_ADDR ; Store pointer to address
53 0501'CF 9E 00D7 451 200$: MOVAB RESERVE_MSG,R3 ; Address of message
01 A3 01 A3 7C 00DC 452 : CLRQ 1(R3) ; Assume no service password (8 bytes)
0000'CF 95 00DF 453 : TSTB PDB$G_CON_SPW ; Did I get a service password?
06 13 00E3 454 : BEQL 300$ ; If EQL no
0001'CF 7D 00E5 455 : MOVQ PDB$G_CON_SPW+1,1(R3) ; String of 8 bytes
52 09 9A 00EB 456 300$: MOVZBL #RESERVE_MSG_LEN,R2 ; Length of message
01BF 30 00EE 457 : BSBW NI_OUTPUT ; Send it
00F1 458 :
00F1 459 : Transmit REQUEST ID message, to see if console is mine
00F1 460 :
53 0046'CF 9E 00F1 461 400$: MOVAB REQUEST_ID_MSG,R3 ; Address of message
52 04 9A 00F6 462 : MOVZBL #REQUEST_MSG_LEN,R2 ; Length of message
01B4 30 00F9 463 : BSBW NI_OUTPUT ; Send it
00FC 464 :
00FC 465 : Receive the System ID message
00FC 466 :
53 0018'CF 9E 00FC 467 : MOVAB SYSTEM_ID_MSG,R3 ; Address of rcv bfr
009C'CF 53 D0 0101 468 : MOVL R3,SYSTEM_ID_END ; Init pointer to end
52 84 8F 9A 0106 469 : MOVZBL #SYSTEM_ID_LTH,R2 ; Length of rcv bfr
018A 30 010A 470 : BSBW NI_INPUT ; Do the receive
24 12 010D 471 : BNEQ 450$ ; If NEQ I/O completion
010F 472 : $CANCEL S NICHAN ; Kill the receive
A7 56 F4 011B 473 410$: SOBGTR R6,150$ ; Try other channel
9F 04AA'CF F5 011E 474 420$: SOBGTR RETRY_COUNTER,100$ ; Try both again
53 0092'CF D0 0123 475 425$: MOVL NO_RESP_MSG+4,R3 ; Address of message
52 008E'CF D0 0128 476 : MOVL NO_RESP_MSG,R2 ; Length of message
02A1 30 012D 477 : BSBW NCP$TTY_WRITE ; Output the message
0135 31 0130 478 : BRW 1500$ ; Done
0133 479 :
0133 480 : Response received
0133 481 :
50 50 10 10 EF 0133 482 450$: EXTZV #16,#16,R0,R0 ; Get length of message
009C'CF 50 C0 0138 483 : ADDL2 R0,SYSTEM_ID_END ; Point to end of message
013D 484 :
013D 485 : See if it is valid
013D 486 :
0018'CF 91 013D 487 : CMPB SYSTEM_ID_MSG,- ; Is this a system ID?
07 07 0141 488 : #MOP$C_SYSTEMID
07 13 0142 489 : BEQL 460$ ; If EQL yes
A8 04AA'CF F5 0144 490 : SOBGTR RETRY_COUNTER,400$ ; See if OK to proceed
D8 11 0149 491 : BRB 425$ ; Ran out of retries
014B 492 :
014B 493 : See if I have the console
014B 494 :
50 02 9A 014B 495 460$: MOVZBL #MOP$C_INFO_FUNC,R0 ; Set to find functions
01B5 30 014E 496 : BSBW SCAN_SYSTEM_ID ; Scan for the field
50 D5 0151 497 : TSTL R0 ; Found?
C6 13 0153 498 : BEQL 410$ ; *** If EQL no - try again
02 51 D1 0155 499 : CMPL R1,#MOP$C_FUNC_SIZE ; Valid size?
```



```
03 62 C1 12 0158 500 BNEQ 410$ ; *** If NEQ no
      05 E1 015A 501 BBC #MOP$V_FUNC_CARR,(R2),470$ ; If BC console is not active
      00A3 31 015E 502 BRW 600$ ; Check reservation
      04AA'CF D7 0161 503 470$: DECL RETRY_COUNTER ; Can I keep going?
      BC 15 0165 505 BLEQ 425$ ; If LEQ no
      0167 506
      0167 507 ; Request downline load of carrier code to DEUNA.
      0167 508
      0167 509 NOTE: We must deassign all our channels to the Ethernet driver in order
      0167 510 to allow MOM/NDDRIVER to be able to issue reads/writes on the protocol
      0167 511 type. XEDRIVER only allows NDDRIVER to be used when NETACP "owns" the
      0167 512 protocol-type UCB, so we must "de-own" it for a while.
      0167 513
      0167 514 $DASSGN_S CHAN = NICHAN_HARD ; Deassign channel to physical address
      0173 515 $DASSGN_S CHAN = NICHAN_PHYS ; Deassign channel to logical address
      017F 516
      5B 0533'CF 9E 017F 517 MOVAB LOAD_PARAMS,R11 ; Set to complete LOAD request msg
      0184 518
      0184 519 ASSUME NMASC_ENT_NOD EQ 0
      0184 520
      8B 0000'CF 90 0184 521 MOVB NCP$GL_OPTION,(R11)+ ; Store load option code
      07 12 0189 522 BNEQ 475$ ; If NEQ load VIA circuit
      018B 523
      018B 524 ASSUME NMASC_ENT_ADD EQ 0
      018B 525
      8B 8B 94 018B 526 CLRB (R11)+ ; Address to follow
      04FF'CF B0 018D 527 MOVW CON$W_ADDR_DNA,(R11)+ ; Store the node address
      50 0000'CF 9E 0192 528 475$: MOVAB PDB$G_CON_PHA,R0 ; Point to address
      80 95 0197 529 TSTB (R0)+ ; Was physical address in command?
      06 13 0199 530 BEQL 480$ ; If EQL no - don't pass one to NML
      8B 0A B0 019B 531 MOVW #NMASC_PCNO_PHA,(R11)+ ; Store PHA indicator
      0222 30 019E 532 BSBW MOVE_IMAGE ; Move it to message
      0000'CF 95 01A1 533 480$: TSTB PDB$G_CON_SLI ; Was service circuit specified?
      0D 13 01A5 534 BEQL 500$ ; If EQL no
      8B 006E 8F B0 01A7 535 MOVW #NMASC_PCNO_SLI,(R11)+ ; Mark service circuit
      50 0001'CF 9E 01AC 536 MOVAB PDB$G_CON_SCI+1,R0 ; Point to circuit string
      020F 30 01B1 537 BSBW MOVE_IMAGE ; Move it to message
      0000'CF 95 01B4 538 500$: TSTB PDB$G_CON_SPW ; Did I get a service password?
      0D 13 01B8 539 BEQL 510$ ; If EQL no
      8B 006F 8F B0 01BA 540 MOVW #NMASC_PCNO_SPA,(R11)+ ; Put in the service password.
      8B 08 90 01BF 541 MOVB #8,(R11)+ ; Set length of service password.
      8B 0001'CF 7D 01C2 542 MOVQ PDB$G_CON_SPW+1,(R11)+ ; Store the service password.
      8B 0078 8F B0 01C7 543 510$: MOVW #NMASC_PCNO_LOA,(R11)+ ; Mark load file
      50 0103'CF 9E 01CC 544 MOVAB LOA_FICE,R0 ; Point to string
      01EF 30 01D1 545 BSBW MOVE_IMAGE ; Move it
      8B 0079 8F B0 01D4 546 MOVW #NMASC_PCNO_SLO,(R11)+ ; Mark secondary loader file
      50 010B'CF 9E 01D9 547 MOVAB SLO_FICE,R0 ; Point to string
      01E2 30 01DE 548 BSBW MOVE_IMAGE ; Move it
      50 0530'CF 9E 01E1 549 MOVAB NICE_LOAD_MSG,R0 ; Point to beginning of message
      0370'CF 9F 01E6 550 PUSHAB CON$LOAD_MSG ; Address of processing routine
      0528'CF 9F 01EA 551 PUSHAB NICE_LOAD_DESC ; Address of message descriptor
      00 BE 5B 50 C3 01EE 552 SUBL3 R0,RT1,a(SP) ; Store message length
      00000000'GF 02 FB 01F3 553 CALLS #2,G^NML$PROCESS,NICE
      0014'DF 91 01FA 554 CMPB @LOAD_RESP_DESC+4,- ; Did the load succeed?
      01 01FE 555 #NMASC_STS_SUC
      52 12 01FF 556 BNEQ 900$ ; If NEQ no
```



```
FE7F 31 0201 557 BRW 10$ ; Try to reserve console again
      0204 558 :
      0204 559 : See if console is reserved
      0204 560 :
      0204 561 600$:
03 62 07 E0 0204 562 BBS #MOP$V_FUNC_RSRV,(R2),700$ ; If BS console is reserved
FF13 31 0208 563 610$: BRW 420$ ; Try reservation again
      0208 564 :
      0208 565 : Console code is active and console is reserved - see if I have it
      0208 566 :
      50 03 9A 0208 567 700$: MOVZBL #MOP$C_INFO_USER,R0 ; Get user
      00F5 30 020E 568 BSBW SCAN_SYSTEM_ID ; Look at message
      50 D5 0211 569 TSTL R0 ; Field present?
      F3 13 0213 570 BEQL 610$ ; If EQL no - try again
62 07EB'CF 06 29 0215 571 CMPC3 #6,MY_ADDRESS,(R2) ; Do I have it?
      10 13 021B 572 BEQL 800$ ; If EQL yes
53 00D8'CF D0 021D 573 MOVL IN_USE_MSG+4,R3 ; Address of message
52 00D4'CF D0 0222 574 MOVL IN_USE_MSG,R2 ; Length of message
      01A7 30 0227 575 BSBW NCP$TTY_WRITE ; Output the message
      003B 31 022A 576 BRW 1500$ ; Finish in common code
      022D 577 :
      022D 578 : Now get to the terminal and start processing characters.
      022D 579 : Also, post a read on the NI channel, for responses
      022D 580 :
53 004F'CF D0 022D 581 800$: MOVL CONNECTED_MSG+4,R3 ; Address of message
52 004B'CF D0 0232 582 MOVL CONNECTED_MSG,R2 ; Length of message
      0197 30 0237 583 BSBW NCP$TTY_WRITE ; Output the message
      023A 584 :
      023A 585 : Set up for terminal loop
      023A 586 :
      01BA 30 023A 587 BSBW TTY_READ ; Start a timed read
      023D 588 $HIBER_S ; Twiddle
      0244 589 :
      0244 590 : If we get here, user typed 'terminate' character
      0244 591 :
53 0088'CF D0 0244 592 MOVL FINISH_MSG+4,R3 ; Address of message
52 0084'CF D0 0249 593 MOVL FINISH_MSG,R2 ; Length of message
      0180 30 024E 594 BSBW NCP$TTY_WRITE ; Output the message
      0A 11 0251 595 BRB 1000$ ; Finish in common code
      0253 596 :
      0253 597 : Output error message from load
      0253 598 :
7E 0010'CF 7D 0253 599 900$: MOVQ LOAD_RESP_DESC,-(SP) ; Set up arguments
0000'CF 02 FB 0258 600 CALLS #2,NCP$CONERR ; Call the error routine
      025D 601 :
      025D 602 : Clean up
      025D 603 :
      025D 604 1000$:
      025D 605 :
      025D 606 : Transmit release message
      025D 607 :
53 004A'CF 9E 025D 608 MOVAB RELEASE_MSG,R3 ; Address of message
52 01 9A 0262 609 MOVZBL #RELEASE_MSG_LEN,R2 ; Length of message
      0048 30 0265 610 BSBW NI_OUTPUT ; Send it
      0268 611 :
      0268 612 : Deassign outstanding channels
      0268 613 :
```

00000000	'GF	00	FB	0268	614	1500\$:	\$DASSGN-S	NICHAN-PHYS	:	Physical address channel
50		00		0274	615		\$DASSGN-S	NICHAN-HARD	:	Hardware address channel
				0280	616	2000\$:	\$DASSGN-S	TTYCHAN	:	Terminal channel
				028C	617		CALLS	-#0,G^NML\$TERMINATE	:	Close off NMLSHR interface
				0293	618		MOVL	S^#SS\$ _NORMAL,RO	:	Status for mainline
				0296	619		RET		:	Done


```
0297 621 :++
0297 622 :
0297 623 : NI_INPUT - This routine receives a message over the Ethernet
0297 624 : NI_OUTPUT - This routine transmits a message over the Ethernet
0297 625 :
0297 626 : Inputs:
0297 627 :     R3 - address of message
0297 628 :     R2 - length of message
0297 629 :
0297 630 : Outputs:
0297 631 :     R0 - first longword of IOSB
0297 632 :     PSL reflects value of R0
0297 633 :
0297 634 : Side effects - image exits if error
0297 635 :
0297 636 :--
0297 637 :
0297 638 NI_INPUT:
0297 639     $SETIMR_S      EFN=#2,-      ; Start timer on input
0297 640     DAYTIM=TIMEOUT
51 50 00' 3C 02A6 641     MOVZWL S^#IOS$ READVBLK,R0      ; Get function code
07EB'CF 9E 02A9 642     MOVAB MY_ADDRESS,R1      ; P5, to get my address
11 11 02AE 643     BRB NI_IO_COMMON      ; Go to common code
02B0 644 NI_OUTPUT:
02B0 645     $CLREF_S      #2      ; Make sure timeout flag isn't set
51 50 00' 3C 02B9 646     MOVZWL S^#IOS$ WRITEVBLK,R0      ; Get function code
04F4'CF D0 02BC 647     MOVL DEST_ADDR,R1      ; P5, to show destination address
02C1 648 NI_IO_COMMON:
02C1 649     $QIO_S      FUNC=R0,CHAN=NICHAN,EFN=#1-
02C1 650     P1=(R3),-
02C1 651     P2=R2,-
02C1 652     P5=R1,-
02C1 653     IOSB=IOSB
02E2 654     ONERROR RET
02E6 655     $WFLOR_S      #1,#^B110      ; Wait for flags 1 or 2
02F1 656     $CANTIM_S      ; Cancel the timer
50 0008'CF D0 02FA 657     MOVL IOSB,R0      ; Get I/O status return
04 13 02FF 658     BEQL 100$      ; If EQL I/O not complete
0301 659     ONERROR RET
05 0305 660 100$: RSB      ; Done
0306 661 :
0306 662 :++
0306 663 :
0306 664 : SCAN_SYSTEM_ID - Scan SYSTEM ID msg for a certain info type
0306 665 :
0306 666 : Message format:
0306 667 :
0306 668 :     Message code - 1 byte
0306 669 :     Reserved byte - 1 byte
0306 670 :     Receipt number - 2 bytes
0306 671 :     Info fields - 2 bytes of type, 1 byte of length, n bytes of data
0306 672 :
0306 673 : Inputs:
0306 674 :     R0 - INFO type field to be matched
0306 675 :
0306 676 : Outputs:
0306 677 :     R0 - 0==> field not found
```

			0306	678	:	R1 - length of info field	
			0306	679	:	R2 - address of info field	
			0306	680	;	--	
			0306	681			
			0306	682		SCAN_SYSTEM ID:	
52	001C	'CF	9E	0306	683	MOVAB	SYSTEM INFO,R2
009C	'CF	52	D1	030B	684	10\$:	CMPL R2,SYSTEM_ID_END
		04	1F	0310	685		BLSSU 15\$
		50	D4	0312	686		CLRL R0
		10	11	0314	687		BRB 100\$
50		82	B1	0316	688	15\$:	CMPL (R2)+,R0
		08	13	0319	689		BEQL 20\$
51		82	9A	031B	690		MOVZBL (R2)+,R1
52		51	C0	031E	691		ADDL2 R1,R2
		E8	11	0321	692		BRB 10\$
51		82	9A	0323	693	20\$:	MOVZBL (R2)+,R1
			05	0326	694	100\$:	RSB

:	Point to start of INFO
:	At end of message?
:	If LSSU no
:	Mark error
:	Take common exit
:	INFO type match?
:	If EQL yes
:	Get length of this field
:	Point to next field
:	Loop
:	Get length of this field
:	Done


```
0327 696 :++
0327 697 :
0327 698 : CON$SHOW MSG - process nice message returned from NMLSHR
0327 699 : This routine extracts the parameters needed for the CONNECT CONSOLE operation
0327 700 : from a NICE message returned by NMLSHR. This routine assumes that there
0327 701 : will be 3 messages: more data, data, done. The 'data' msg is of interest.
0327 702 :
0327 703 : CALLS #1,CON$SHOW_MSG
0327 704 :
0327 705 : Input:
0327 706 :
0327 707 : 4(AP) - Address of descriptor of message
0327 708 :
0327 709 :--
0327 710 :
0327 711 CON$SHOW MSG: .WORD ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
51 04 BC 7D 0329 712 MOVQ @4(AP),R1 ; Get message descriptor
02 62 91 032D 713 CMPB (R2),#NMASC_STS_MOR ; Do I want to see this?
3D 13 0330 714 BEQL 100$ ; If EQL no
80 8F 62 91 0332 715 CMPB (R2),#NMASC_STS_DON ; Do I want to see this?
37 13 0336 716 BEQL 100$ ; If EQL no
01 62 91 0338 717 CMPB (R2),#NMASC_STS_SUC ; Do I want to see this?
0E 13 033B 718 BEQL 5$ ; If EQL yes
07E1'CF 96 033D 719 INCB NICE_MSG_FLAG ; Set error flag
7E 51 7D 0341 720 MOVQ R1,-(SP) ; Set up for call to error processor
0000'CF 02 FB 0344 721 CALLS #2,NCP$CONERR ; Output error
24 11 0349 722 BRB 100$ ; Return
034B 723 :
034B 724 : Initialize to scan for missing parameters
034B 725 :
5A 0040'CF 9E 034B 726 5$: MOVAB NICE_PARAM_ID,R10 ; Point to data IDs
59 0030'CF 9E 0350 727 MOVAB NICE_PARAM_ADDR,R9 ; Point to data storage addresses
0355 728 :
0355 729 : Loop to get parameters not specified in command
0355 730 :
50 8A 3C 0355 731 10$: MOVZWL (R10)+,R0 ; Get param type code
5B 89 D0 0358 732 MOVL (R9)+,R11 ; Get PDB address
12 13 035B 733 BEQL 100$ ; If EQL done
8B 95 035D 734 TSTB (R11)+ ; Was param passed?
F4 12 035F 735 BNEQ 10$ ; If NEQ yes
5B DD 0361 736 PUSHL R11 ; Address of storage area to stack
50 DD 0363 737 PUSHL R0 ; Param type code to stack
0000'CF 02 FB 0365 738 CALLS #2,CON$PARSE_NICE ; Parse the field, if it's there
7B 50 90 036A 739 MOVB R0,-(R11) ; Return value becomes existence flag
E6 11 036D 740 BRB 10$ ; Loop
04 036F 741
036F 742 100$: RET ; Done
```

```

0370 744 :++
0370 745 :
0370 746 : CONSLOAD_MSG - process response to load message returned from NMLSHR
0370 747 :
0370 748 : CALLS #1,CONSLOAD_MSG
0370 749 :
0370 750 : Input:
0370 751 :
0370 752 : 4(AP) - Address of descriptor of message
0370 753 :
0370 754 :--
0370 755 :
0010'CF 04 BC OFFC 0370 756 CONSLOAD_MSG: .WORD *M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
          7D 0372 757 MOVQ @4(AP),LOAD_RESP_DESC ; Get message descriptor
          04 0378 758 100$: RET ; Done

```



```
0379 760 :++
0379 761 :
0379 762 : SETUP_NICHAN - Set up channel to Ethernet
0379 763 :
0379 764 : INPUTS:
0379 765 :
0379 766 :         R2 - Address of Ethernet address for destination on this channel
0379 767 :         R6 - Address of channel no.
0379 768 :
0379 769 : OUTPUTS:
0379 770 :
0379 771 :         Channel number stored indirectly via R6
0379 772 :
0379 773 :--
0379 774 :
0379 775 : SETUP_NICHAN:
04F4'CF 62 06 28 0379 776 :         MOVCL #6,(R2),DEST_ADDR ; Stuff the address
037F 777 :
037F 778 :         Get a channel to Ethernet
037F 779 :
037F 780 :         $ASSIGN_S DEVNAM = @CON$XE_DESCADDR, CHAN = (R6)
038E 781 :         ONERROR RET ; If error done
0392 782 :
0392 783 :         Set function to establish the protocol type and start device.
0392 784 :         The initial startup will take about 6 seconds for the DEUNA
0392 785 :         to run the self-test sequence.
0392 786 :
0392 787 :         $QIOW_S FUNC = #<IOS$ SETMODE!IOSM_CTRL!IOSM_STARTUP>,-
0392 788 :         CHAN = (R6),-
0392 789 :         IOSB = IOSB,-
0392 790 :         P2 = #SETPARAM_DESC
0385 791 :         ONERROR RET ; If error done
50 0008'CF D0 0389 792 :         MOVL IOSB, R0 ; Else, get I/O status return
038E 793 :         ONERROR RET ; If error done
05 03C2 794 :         RSB ; Done
```

```

03C3 796 :++
03C3 797 :
03C3 798 : MOVE_IMAGE - move image string
03C3 799 :
03C3 800 : INPUTS:
03C3 801 :
03C3 802 :         R0 - address of source string
03C3 803 :         R11 - address of destination string
03C3 804 :
03C3 805 : OUTPUTS:
03C3 806 :
03C3 807 :         R0-R5 - clobbered
03C3 808 :         R11 - updated to point at next storage byte
03C3 809 :
03C3 810 : --
03C3 811 :
03C3 812 : MOVE_IMAGE:
03C3 813 :     MOVZBL (R0)+,R1 ; Get length
03C3 814 :     MOVB R1,(R11)+ ; Store it
03C3 815 :     MOVC3 R1,(R0),(R11) ; Store string
03C3 816 :     MOVL R3,R11 ; Update pointer
03C3 817 :     RSB ; Done

```

68 51 80 9A 03C3 813 MOVZBL (R0)+,R1 ; Get length
 8B 51 90 03C6 814 MOVB R1,(R11)+ ; Store it
 60 51 28 03C9 815 MOVC3 R1,(R0),(R11) ; Store string
 5B 53 D0 03CD 816 MOVL R3,R11 ; Update pointer
 05 03D0 817 RSB ; Done


```
03D1 819 :++
03D1 820 :
03D1 821 : NCP$TTY_WRITE - This routine writes a message on the terminal
03D1 822 :
03D1 823 : INPUTS:
03D1 824 :
03D1 825 :     R2 - length of msg
03D1 826 :     R3 - address of msg
03D1 827 :
03D1 828 :--
03D1 829 :
03D1 830 NCP$TTY_WRITE::
03D1 831     $QIOW_S CHAN = TTYCHAN,- ; Get input from terminal
03D1 832     FUNC = #<IOS_WRITEVBLK>,-
03D1 833     IOSB = IOSB,-
03D1 834     P1 = (R3),-
03D1 835     P2 = R2
03F2 836
03F2 837     ONERROR RET
05 03F6 838     RSB ; Done
03F7 839
03F7 840
03F7 841 :++
03F7 842 :
03F7 843 : TTY_READ - Routine to do timed read from terminal. Also, input
03F7 844 : terminates on BREAK_CHAR (and "break" indication is sent across channel) or
03F7 845 : DONE_CHAR, which results in termination of operation. I/O completes in an
03F7 846 : AST, where input is processed and another I/O is queued.
03F7 847 :
03F7 848 :--
03F7 849 :
03F7 850 TTY_READ:
50 0000'CF 9E 03F7 851     MOVAB TERMINATOR_DESC,R0 ; Get address of character mask info
03FC 852     $QIO_S CHAN = TTYCHAN,- ; Get input from terminal
03FC 853     FUNC = #<IOS_TTYREADALL!IOSM_TIMED!IOSM_NOECHO>,-
03FC 854     IOSB = TTY_IN_IOSB,-
03FC 855     ASTADR = TTY_AST,-
03FC 856     P1 = TTY_INBUF,-
03FC 857     P2 = #XMTBUFLN,-
03FC 858     P3 = #1,- ; 1 second timer
03FC 859     P4 = R0 ; Descriptor of terminator set
0429 860
0429 861     ONERROR RET
05 042D 862     RSB ; Done
042E 863
042E 864
042E 865 :++
042E 866 :
042E 867 : TTY_AST - Process terminal input ASTs
042E 868 : This routine is called when a timeout on the TTY read occurs or when
042E 869 : one of the terminators is encountered. If a timeout occurs, any characters
042E 870 : in the buffer are sent in a COMMAND/POLL message. If the "break" terminator
042E 871 : is encountered, any characters in the terminal buffer are sent, and the
042E 872 : "break" indicator is set in the message. If the "done" terminator is
042E 873 : encountered, the process is restarted, so that control will return to mainline.
042E 874 : The NI input buffer is also checked for completion at this point; if there
042E 875 :
```

```
042E 876 ; is a message, the text is printed on the terminal and another NI read is
042E 877 ; queued.
042E 878 ;
042E 879 ;--
042E 880
042E 881 TTY_AST: .WORD ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
0430 882 BICB #MOP$M_BREAK,MOP_CTRL_FLAGS ; Assume no break
0435 883 BLBC TTY_IN_IOSB,100$ ; If LBC assume timeout occurred
043A 884 CMPW TTY_IN_IOSB+4,#DONE_CHAR ; Was that the I/O terminator?
043F 885 BNEQ 50$ ; If NEQ no
0441 886 10$: $WAKE_S ; We are all done
044C 887 BRB 200$ ; Finish in common code
044E 888
044E 889 50$: CMPW TTY_IN_IOSB+4,#BREAK_CHAR ; Was that the I/O terminator?
0453 890 BNEQ 100$ ; If NEQ no
0455 891
0455 892 BISB #MOP$M_BREAK,MOP_CTRL_FLAGS ; Set break indicator
045A 893 ;
045A 894 ; Set up to transmit to remote console
045A 895
045A 896 100$: BICB2 #MOP$M_MSGNUM,MOP_CTRL_FLAGS ; Clear msg number
045F 897 MCOMB MOP_MSG_NUM,MOP_MSG_NUM ; Get next number-1
0466 898 ADDB3 #1,MOP_MSG_NUM,R0 ; Get bit to put in message
046C 899 BEQL 110$ ; If EQL number is right
046E 900 INCB MOP_CTRL_FLAGS ; Make the number right
0472 901 110$: MOVAB NI_XMTBUF,R3 ; Get address of output buffer
0477 902 MOVZWL TTY_IN_IOSB+2,R2 ; Get number of characters in
047C 903 ADDL2 #COMMAND_OHD,R2 ; Account for rest of message
047F 904 BSBW NI_OUTPUT ; Ship the message
0482 905 MOVAB NI_RCVBUF,R3 ; Address of rcv bfr
0487 906 MOVZWL #RCVBUFLN,R2 ; Length of rcv bfr
048C 907 BSBW NI_INPUT ; Do the receive
048F 908 BNEQ 120$ ; If NEQ I/O completion
0491 909 MOVL NO_RESP_MSG+4,R3 ; Address of message
0496 910 MOVL NO_RESP_MSG,R2 ; Length of message
049B 911 BSBW NCP$TTY_WRITE ; Output the message
049E 912 BRB 10$ ; Done
04A0 913 ;
04A0 914 ; Terminal output from console
04A0 915 ;
04A0 916 120$: MOVAB NI_RCVBUF,R3 ; Point to message
04A5 917 CMPB (R3)+,#MOP$C_RESPONSE ; Is this a response message
04A8 918 BNEQ 190$ ; If NEQ no - ignore
04AA 919 MOVAB (R3)+,R0 ; Get the control flags
04AD 920 BBS #MOP$V_CMDLOST,R0,110$ ; If BS need to rexmt
04B1 921 MOVZWL IOSB+2,R2 ; Get no. of chars in message
04B6 922 SUBL2 #2,R2 ; Adjust for MOP overhead
04B9 923 BSBW NCP$TTY_WRITE ; Output to the terminal
04BC 924 190$: BSBW TTY_READ ; Start the next input
04BF 925 200$: RET ; Done
04C0 926
04C0 927 .END
```


NCPCONCAR
Symbol table

N 7

- Console Carrier Requester Ethernet I/O 16-SEP-1984 01:58:40 VAX/VMS Macro V04-00
5-SEP-1984 02:13:29 [NCP.SRC]NCPCONCAR.MAR;1

Page 24
(1)

```

$ST1 = 00000000
ACTSVRB CONNECT = 00000000 RG 04
ADDR_HARD 000007E3 R 03
ADDR_PHYS 000004FA R 03
BREAK_CHAR = 00000002
COMMAND_OHD = 00000002
CONSGET_SLI ***** X 04
CONSLOAD_MSG 00000370 R 04
CONSNUFB 0000056F RG 03
CONSNUFB_DESC 00000008 RG 02
CONSP2BUF 0000058B RG 03
CONSP2BUF_DESC 00000010 RG 02
CONSP4BUF 000005E1 RG 03
CONSP4BUF_DESC 00000018 RG 02
CONSPARSE_NICE ***** X 04
CONSSHOW_MSG 00000327 R 04
CONST_SHOW_NODE 00000521 RG 03
CONSW_ADDR_DNA 000004FF RG 03
CONSEX_DESCADDR 000004A6 RG 03
CONNECTED_MSG 0000004B R 02
CR = 0000000D
DEST_ADDR = 000004F4 R 03
DONE_CHAR = 00000004
FINISH_MSG 00000084 R 02
IN_USE_MSG 000000D4 R 02
IOSM_CTRL ***** X 04
IOSM_NOECHO ***** X 04
IOSM_STARTUP ***** X 04
IOSM_TIMED ***** X 04
IOS_READVBLK ***** X 04
IOS_SETMODE ***** X 04
IOS_TTYREADALL ***** X 04
IOS_WRITEVBLK ***** X 04
IOSB 00000008 R 03
LF = 0000000A
LOAD_PARAMS 00000533 R 03
LOAD_RESP_DESC 00000010 R 03
LOA_FILE 00000103 R 02
LOA_LTH = 00000008
MOPSC_COMMAND = 00000011
MOPSC_FUNC_SIZE = 00000002
MOPSC_INFO_CSIZ = 00000005
MOPSC_INFO_FUNC = 00000002
MOPSC_INFO_RSIZ = 00000006
MOPSC_INFO_TIMR = 00000004
MOPSC_INFO_USER = 00000003
MOPSC_RELEASE = 0000000F
MOPSC_REQID = 00000005
MOPSC_RESERVE = 0000000D
MOPSC_RESPONSE = 00000013
MOPSC_SYSTEMID = 00000007
MOPSM_BREAK = 00000002
MOPSM_MSGNUM = 00000001
MOPSV_CMDLOST = 00000001
MOPSV_FUNC_CARR = 00000005
MOPSV_FUNC_RSRV = 00000007
MOP_CTRL_FLAGS 000002A5 R 03

```

```

MOP_MSG_NUM 000007E2 R 03
MOVE_IMAGE 000003C3 R 04
MY_ADDRESS 000007EB R 03
NCPSCONERR ***** X 04
NCPSTTY_WRITE ***** X 04
NETMANVER 000003D1 RG 04
NFBSC_CTX_SIZE 00000100 R 02
NFBSC_LENGTH = 00000040
NFBSC_SIZE = 00000010
NICE_LOAD_DESC 00000528 R 03
NICE_LOAD_MSG 00000530 R 03
NICE_MSG_FLAG 000007E1 R 03
NICE_PARAM_ADDR 00000030 R 02
NICE_PARAM_ID 00000040 R 02
NICE_SHOW_DESC 00000517 R 03
NICE_SHOW_MSG 0000051F R 03
NICHAN 000004BA R 03
NICHAN_HARD 000004B6 R 03
NICHAN_PHYS 000004B8 R 03
NI_ADDRESSES 000004AE R 03
NI_INPUT 00000297 R 04
NI_IO_COMMON 000002C1 R 04
NI_OUTPUT 000002B0 R 04
NI_RCVBUF 000000A0 R 03
NI_XMTBUF 000002A4 R 03
NMASC_ACC_LIM = 00000002
NMASC_ENT_ADD = 00000000
NMASC_ENT_NOD = 00000000
NMASC_FNC_LOA = 0000000F
NMASC_FNC_REA = 00000014
NMASC_FNC_SYS = 00000016
NMASC_LINMC_SET = 00000001
NMASC_OPINF_CHA = 00000002
NMASC_PCLI_ACC = 00000B1E
NMASC_PCLI_BFN = 00000451
NMASC_PCLI_BUS = 00000AF1
NMASC_PCLI_CRC = 00000B1C
NMASC_PCLI_DCH = 00000B1B
NMASC_PCLI_DES = 00000B21
NMASC_PCLI_PAD = 00000B1A
NMASC_PCLI_PRN = 00000B18
NMASC_PCLI_PTY = 00000B0E
NMASC_PCNO_HWA = 00000072
NMASC_PCNO_LOA = 00000078
NMASC_PCNO_PHA = 0000000A
NMASC_PCNO_SLI = 0000006E
NMASC_PCNO_SLO = 00000079
NMASC_PCNO_SPA = 0000006F
NMASC_STATE_OFF = 00000001
NMASC_STATE_ON = 00000000
NMASC_STS_DON = FFFFFFFF80
NMASC_STS_MOR = 00000002
NMASC_STS_SUC = 00000001
NMASC_SYS_VMS = 00000004
NMLSINITIALIZE ***** X 04
NMLSPROCESS_NICE ***** X 04

```

NCPCONCAR
Symbol table

B 8

- Console Carrier Requester Ethernet I/O 16-SEP-1984 01:58:40 VAX/VMS Macro V04-00
5-SEP-1984 02:13:29 [NCP.SRC]NCPCONCAR.MAR;1

Page 25
(1)

NML\$TERMINATE	*****	X	04
NO_HARD_MSG	00000080	R	02
NO_RESP_MSG	0000008E	R	02
P2BUFSIZ	= 00000056		
P4BUFSIZ	= 00000200		
PDB\$G_CON_PHA	*****	X	04
PDB\$G_CON_SLI	*****	X	02
PDB\$G_CON_SPW	*****	X	02
PDB\$G_VRB_ENT	*****	X	04
RCVBUFLN	= 00000200		
RELEASE_MSG	0000004A	R	02
RELEASE_MSG_LEN	= 00000001		
REQUEST_ID_MSG	00000046	R	02
REQUEST_MSG_LEN	= 00000004		
RESERVE_MSG	00000501	R	03
RESERVE_MSG_LEN	= 00000009		
RETRY_CNT	= 00000005		
RETRY_COUNTER	000004AA	R	03
SCAN_SYSTEM_ID	00000306	R	04
SETPARAM	000004BE	R	03
SETPARAMLEN	= 0000003C		
SETPARAM_DESC	00000020	R	02
SETUP_NICHAN	00000379	R	04
SLO_FILE	0000010B	R	02
SLO_LTH	= 00000008		
SS\$NORMAL	*****	X	04
SY\$ASSIGN	*****	GX	04
SY\$CANCEL	*****	GX	04
SY\$CANTIM	*****	GX	04
SY\$CLREF	*****	GX	04
SY\$DASSGN	*****	GX	04
SY\$HIBER	*****	GX	04
SY\$QIO	*****	GX	04
SY\$QIOW	*****	GX	04
SY\$SETIMR	*****	GX	04
SY\$WAKE	*****	GX	04
SY\$WFLOR	*****	GX	04
SYSTEM_ID_END	0000009C	R	03
SYSTEM_ID_LTH	= 00000084		
SYSTEM_ID_MSG	00000018	R	03
SYSTEM_INFO	= 0000001C	R	03
TERMINATOR_DESC	00000000	R	02
TIMEOUT	00000028	R	02
TIMER	= 00000005		
TTYCHAN	000004BC	R	03
TTY_AST	0000042E	R	04
TTY_DESC	000000ED	R	02
TTY_INBUF	= 000002A6	R	03
TTY_IN_IOSB	00000000	R	03
TTY_READ	000003F7	R	04
XE_DESC	0000050A	R	03
XMTBUFLN	= 00000200		
ZERO_LTH	= 0000027C		
ZERO_START	0000056F	R	03

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

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
\$AB\$\$	00000000 (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
CONCAR\$PURE	00000113 (275.)	02 (2.)	NOPIC USR CON REL LCL NOSHR NOEXE RD NOWRT NOVEC BYTE
CONCAR\$IMPURE	000007F9 (2041.)	03 (3.)	NOPIC USR CON REL LCL NOSHR NOEXE RD WRT NOVEC LONG
CONCAR\$CODE	000004C0 (1216.)	04 (4.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC BYTE

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

Phase	Page faults	CPU Time	Elapsed Time
Initialization	32	00:00:00.09	00:00:00.93
Command processing	132	00:00:00.84	00:00:07.60
Pass 1	426	00:00:14.84	00:00:49.81
Symbol table sort	0	00:00:02.07	00:00:05.82
Pass 2	173	00:00:03.19	00:00:08.92
Symbol table output	21	00:00:00.18	00:00:00.57
Psect synopsis output	3	00:00:00.03	00:00:00.16
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	789	00:00:21.25	00:01:13.81

The working set limit was 1800 pages.
83239 bytes (163 pages) of virtual memory were used to buffer the intermediate code.
There were 90 pages of symbol table space allocated to hold 1503 non-local and 63 local symbols.
927 source lines were read in Pass 1, producing 23 object records in Pass 2.
26 pages of virtual memory were used to define 25 macros.

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

Macro library name	Macros defined
_\$255\$DUA28:[SHRLIB]NET.MLB;1	1
_\$255\$DUA28:[SHRLIB]NMALIBRY.MLB;1	1
_\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	0
_\$255\$DUA28:[SYSLIB]STARLET.MLB;2	19
TOTALS (all libraries)	21

1560 GETS were required to define 21 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS\$:NCPCONCAR/OBJ=OBJ\$:NCPCONCAR MSRC\$:NCPCONCAR/UPDATE=(ENH\$:NCPCONCAR)+EXECML\$/LIB+SHRLIB\$:NMALIBRY/LIB+SHRLIB\$:NET/LIB

0267 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

NCPCONCAR
LIS

NCPERRMSG
LIS

NCPCONMAN
LIS

NCPLIBRY
B32

NCPMAIN
LIS

NCPNETIO
LIS

NMAHEAD
B32

NCPLIBRY
LIS

NMATAIL
B32