

NNN		NNN	MMM	MMM	LLL
NNN		NNN	MMM	MMM	LLL
NNN		NNN	MMM	MMM	LLL
NNN		NNN	MMMMMM	MMMMMM	LLL
NNN		NNN	MMMMMM	MMMMMM	LLL
NNN		NNN	MMMMMM	MMMMMM	LLL
NNNNNN		NNN	MMM	MMM	LLL
NNNNNN		NNN	MMM	MMM	LLL
NNNNNN		NNN	MMM	MMM	LLL
NNN	NNN	NNN	MMM	MMM	LLL
NNN	NNN	NNN	MMM	MMM	LLL
NNN	NNN	NNN	MMM	MMM	LLL
NNN	NNNNNN	NNN	MMM	MMM	LLL
NNN	NNNNNN	NNN	MMM	MMM	LLL
NNN	NNNNNN	NNN	MMM	MMM	LLL
NNN	NNNNNN	NNN	MMM	MMM	LLL
NNN	NNN	NNN	MMM	MMM	LLL
NNN	NNN	NNN	MMM	MMM	LLL
NNN	NNN	NNN	MMM	MMM	LLL
NNN	NNN	NNN	MMM	MMM	LLLLLLLLLLLLLLLL
NNN	NNN	NNN	MMM	MMM	LLLLLLLLLLLLLLLL
NNN	NNN	NNN	MMM	MMM	LLLLLLLLLLLLLLLL

_S

Ps

NP

NP

\$G

\$O

NP

PA

_L

```

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
LLLLLLLLLLLL           IIIIII                      SSSSSSSS
LLLLLLLLLLLL           IIIIII                      SSSSSSSS

```



```
0001 0 %TITLE 'Network Management Listener main module'
0002 0 MODULE NML$MAIN (MAIN = NML$MAIN,
0003 0 ADDRESSING_MODE (NONEXTERNAL=GENERAL),
0004 0 ADDRESSING_MODE (EXTERNAL=GENERAL),
0005 0 IDENT = 'V04-000') =
0006 0
0007 1 BEGIN
0008 1
0009 1
0010 1 *****
0011 1 *
0012 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0013 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0014 1 * ALL RIGHTS RESERVED.
0015 1 *
0016 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0017 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0018 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0019 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0020 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0021 1 * TRANSFERRED.
0022 1 *
0023 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0024 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0025 1 * CORPORATION.
0026 1 *
0027 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0028 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0029 1 *
0030 1 *
0031 1 *****
0032 1
0033 1
0034 1 ++
0035 1 FACILITY: DECnet-VAX V2.0 Network Management Listener
0036 1
0037 1 ABSTRACT:
0038 1
0039 1 This is the main module for NML.
0040 1
0041 1 ENVIRONMENT: VAX/VMS Operating System
0042 1
0043 1 AUTHOR: Distributed Systems Software Engineering
0044 1
0045 1 CREATION DATE: 1-OCT-1979
0046 1
0047 1 MODIFIED BY:
0048 1
0049 1 V03-004 MKP0005 Kathy Perko 3-Jan-1984
0050 1 Convert old $TRNLOG to new $TRNLNM system service for
0051 1 translating logical names.
0052 1
0053 1 V03-003 MKP0004 Kathy Perko 21-April-1983
0054 1 Delete service functions from NML.
0055 1
0056 1 V03-002 MKP0003 Kathy Perko 1-March-1983
0057 1 Get rid of extra NET: channel to NETACP which was used
```



```

: 58      0058 1  |
: 59      0059 1  |
: 60      0060 1  |
: 61      0061 1  |
: 62      0062 1  |
: 63      0063 1  |
: 64      0064 1  |
: 65      0065 1  |
: 66      0066 1  |
: 67      0067 1  |
: 68      0068 1  |
: 69      0069 1  |
: 70      0070 1  |
: 71      0071 1  |
: 72      0072 1  |
: 73      0073 1  |
: 74      0074 1  |
: 75      0075 1  |
: 76      0076 1  |
: 77      0077 1  |
: 78      0078 1  |
: 79      0079 1  |
: 80      0080 1  |
: 81      0081 1  |
: 82      0082 1  |
: 83      0083 1  |
: 84      0084 1  |
: 85      0085 1  |

```

when NML was part the network booting process.

V03-001 MKP0002 Kathy Perko 17-Nov-1982
Add checking for Network Management Version 4.0.0 and
consolidate the version checking code into one routine.

V005 MKP0001 Kathy Perko 17-Dec-1981
Bypass version number checking for autoservice functions.

V004 TMH0004 Tim Halvorsen 12-Oct-1981
Store NICE version number of NCP we are communicating
with, so that some level of V2.0 compatibility can be done.
Mark Phase II (missing version number) as V1.3.0 to
NML\$INITIALIZE routine.

V003 MKP0003 Kathy Perko 03-Sep-1981
Temporarily tell 2.0 NCP's that NML is 2.0 also.

V002 TMH0002 Tim Halvorsen 31-Aug-1981
Allow communication with 2.0.0 NCPs.

V001 TMH0001 Tim Halvorsen 20-Jul-1981
Use sharable image callable interface.
Make program exit quietly when the link to NCP is aborted.
Remove network initialization code, since NCP no longer
initiates NML as a detached process during startup.


```

: 87      0086 1 %SBTTL 'Declarations'
: 88      0087 1
: 89      0088 1
: 90      0089 1  TABLE OF CONTENTS:
: 91      0090 1
: 92      0091 1
: 93      0092 1  FORWARD ROUTINE
: 94      0093 1      NML$MAIN          : NOVALUE,
: 95      0094 1      NML_COMMANDS      : NOVALUE,
: 96      0095 1      NML_INITIALIZE,
: 97      0096 1      NML_CONNECT,
: 98      0097 1      NML_VALIDLINK,
: 99      0098 1      NML_GETMODE       : NOVALUE,
100      0099 1      NML_RECEIVE,
101      0100 1      NML_RESPONSE:      NOVALUE;      ! Receive NICE message over link
102      0101 1
103      0102 1  INCLUDE FILES:
104      0103 1
105      0104 1
106      0105 1
107      0106 1  LIBRARY 'LIB$:NMLLIB';      ! Facility-wide definitions
108      0107 1  LIBRARY 'SHRLIB$:NMLIBRY';  ! NICE definitions
109      0108 1  LIBRARY 'SHRLIB$:NET';      ! NETACP QIO interface
110      0109 1  LIBRARY 'SYS$LIBRARY:STARLET'; ! VMS common definitions
111      0110 1
112      0111 1
113      0112 1  NML mode symbols (set in NML$B_MODE)
114      0113 1
115      0114 1
116      0115 1  LITERAL
117      0116 1      NML$_INIT      = 0,      ! Initializing
118      0117 1      NML$_CONNECT = 1;      ! Started via connect
119      0118 1
120      0119 1
121      0120 1  OWN STORAGE:
122      0121 1
123      0122 1  BIND
124      0123 1      NML_VERSION = UPLIT BYTE (NML$K_VERSION,
125      0124 1      NML$K_DEC_ECO,
126      0125 1      NML$K_USER_ECO);
127      0126 1
128      0127 1  OWN
129      0128 1      NML$B_MODE      : BYTE
130      0129 1      INITIAL (NML$_INIT), ! NML mode of operation
131      0130 1      NML$L_CMDNCBLEN, ! Length of NCB data
132      0131 1      NML$W_CMD_CHAN: WORD, ! Channel to logical link to NCP
133      0132 1      NML$A_NCBADDRESS, ! Address of NCB
134      0133 1      NML$A_NCBDATA,    ! Address of optional NCB data field
135      0134 1      NML$Q_SRVIDDSC: BLOCK [8,BYTE], ! Descriptor of autoservice NCB
136      0135 1      NML$AB_RCVBUFFER: VECTOR [nml$k_rcvbflen, BYTE]; ! Message buffer
137      0136 1
138      0137 1
139      0138 1  EXTERNAL REFERENCES:
140      0139 1
141      0140 1
142      0141 1  EXTERNAL ROUTINE
143      0142 1      NML$INITIALIZE,      ! Initialize NICE processor
```

NML\$MAIN
V04-000

Network Management Listener main module
Declarations

: 144
: 145
: 146

0143 1 NML\$PROCESS_NICE: NOVALUE,
0144 1 NML\$TERMINATE: NOVALUE,
0145 1 LIB\$ASN_WTH_MBX;

I 13
15-Sep-1984 23:57:11 VAX-11 Bliss-32 V4.0-742
14-Sep-1984 12:50:12
! Process a single NICE message
! Terminate NICE processor
! Assign channel with mailbox

DISK\$VMSMASTER:[NML.SRC]NMLMAIN.B32;1 Page 4
(2)

NM
VO


```
: 148 0146 1 %SBTTL 'NML$MAIN Main routine'
: 149 0147 1 ROUTINE NML$MAIN : NOVALUE =
: 150 0148 1
: 151 0149 1 !++
: 152 0150 1 ! FUNCTIONAL DESCRIPTION:
: 153 0151 1
: 154 0152 1 ! This is the main routine for the DECnet-VAX Network Management
: 155 0153 1 ! Listener (NML).
: 156 0154 1
: 157 0155 1 ! ROUTINE VALUE:
: 158 0156 1 ! COMPLETION CODES:
: 159 0157 1
: 160 0158 1 ! Always returns NML$_STS_SUC.
: 161 0159 1
: 162 0160 1 !--
: 163 0161 1
: 164 0162 2 BEGIN
: 165 0163 2
: 166 0164 2 !
: 167 0165 2 ! Determine how we were initiated.
: 168 0166 2
: 169 0167 2 IF NML_INITIALIZE () THEN
: 170 0168 2 BEGIN
: 171 0169 3 NML_COMMANDS (); ! NICE command mode
: 172 0170 3 NML$TERMINATE(); ! Terminate NICE processor
: 173 0171 2 END;
: 174 0172 2
: 175 0173 2 $EXIT (CODE = TRUE);
: 176 0174 2
: 177 0175 1 END; ! End of NML
```

```
.TITLE NML$MAIN Network Management Listener main modul
```

```
.IDENT \V04-000\
```

```
.PSECT $PLITS$,NOWRT,NOEXE,2
```

```
00 00 04 00000 P.AAA: .BYTE 4, 0, 0 ;
```

```
.PSECT $OWNS$,NOEXE,2
```

```
00 00000 NML$B_MODE:
      .BYTE 0
00001 .BLKB 3
00004 NML$C_CMDNCBLEN:
      .BLKB 4
00008 NML$W_CMD_CHAN:
      .BLKB 2
0000A .BLKB 2
0000C NML$A_NCBADDRESS:
      .BLKB 4
00010 NML$A_NCBDATA:
      .BLKB 4
00014 NML$Q_SRVIDDSC:
      .BLKB 8
0001C NML$AB_RCVBUFFER:
```

.BLKB 512

NML_VERSION=
P.AAA

.EXTRN NML\$INITIALIZE, NML\$PROCESS NICE
.EXTRN NML\$TERMINATE, LIB\$ASN_WTH_MBX
.EXTRN SYS\$EXIT

.PSECT \$CODE\$,NOWRT,2

0000 00000 NML\$MAIN:

00000000V 00
00000000V 00
00000000G 00
00000000G 00

00 FB 00002
50 E9 00009
00 FB 0000C
00 FB 00013
01 DD 0001A 1\$:
01 FB 0001C
04 00023

.WORD Save nothing
CALLS #0, NML_INITIALIZE
BLBC R0, 1\$
CALLS #0, NML_COMMANDS
CALLS #0, NML\$TERMINATE
PUSHL #1
CALLS #1, SYS\$EXIT
RET

: 0147
: 0167
: 0169
: 0170
: 0173
: 0175

; Routine Size: 36 bytes, Routine Base: \$CODE\$ + 0000


```
179 0176 1 %SBTTL 'NML_COMMANDS Main command processing routine'
180 0177 1 ROUTINE NML_COMMANDS: NOVALUE =
181 0178 1
182 0179 1 !++
183 0180 1 FUNCTIONAL DESCRIPTION:
184 0181 1
185 0182 1 This routine is the main command processing routine. NICE messages
186 0183 1 are parsed to determine the requested function and then the proper
187 0184 1 routine is called to perform the function.
188 0185 1
189 0186 1 !--
190 0187 1
191 0188 2 BEGIN
192 0189 2
193 0190 2 LOCAL
194 0191 2 desc: VECTOR [2], ! Message descriptor
195 0192 2 status; ! Temporary status
196 0193 2
197 0194 2
198 0195 2 Receive NICE messages and perform valid functions as long as the link
199 0196 2 is active.
200 0197 2
201 0198 2
202 0199 2 WHILE TRUE
203 0200 2 DO
204 0201 2 BEGIN
205 0202 2 desc [1] = nml$ab_rcvbuffer;
206 0203 2 status = nml_receive (.desc [1],
207 0204 2 nml$k_rcvbflen,
208 0205 2 desc [0]);
209 0206 2
210 0207 2 IF .status
211 0208 2 THEN
212 0209 2 nml$process_nice(desc, nml_response)
213 0210 2
214 0211 2 If the message received was too large for the buffer then send a status
215 0212 2 message indicating the error. Any other errors indicate that the link
216 0213 2 has been broken and NML should exit.
217 0214 2
218 0215 2
219 0216 2 ELSE
220 0217 2 BEGIN
221 0218 2 IF .STATUS EQLU SS$_LINKABORT ! If normal exit (via link DASSGN
222 0219 2 OR .STATUS EQLU SS$_LINKDISCON ! or formal disconnect)
223 0220 2 THEN
224 0221 2 RETURN; ! then exit successfully
225 0222 2
226 0223 2 IF .STATUS EQLU SS$_DATAOVERUN ! If message was too large,
227 0224 2 THEN
228 0225 2 NML_RESPONSE(UPLIT(1, UPLIT BYTE(NMASC_STS_SIZ))) ! Send size error
229 0226 2 ELSE
230 0227 2 SIGNAL_STOP (.STATUS); ! Signal fatal error
231 0228 2 END;
232 0229 2
233 0230 2 END;
234 0231 2
235 0232 1 END;
```

```

                                .PSECT $SPLIT$,NOWRT,NOEXE,2
                                FC 00003 P.AAC: .BYTE -4
                                00000001, 00004 P.AAB: .LONG 1
                                00000000, 00008 .ADDRESS P.AAC

                                .PSECT $CODE$,NOWRT,2
                                000C 00000 NML_COMMANDS:
                                .WORD Save R2,R3
                                MOVAB NML_RESPONSE, R3
                                SUBL2 #8, SP
                                MOVAB NML$AB_RCVBUFFER, DESC+4
                                PUSHL SP
                                MOVZWL #512, -(SP)
                                PUSHL DESC+4
                                CALLS #3, NML_RECEIVE
                                MOVL R0, STATUS
                                BLBC STATUS, 2$
                                PUSHL R3
                                PUSHAB DESC
                                CALLS #2, NML$PROCESS_NICE
                                BRB 1$
                                00000000V 00 9E 00002 1$: CMPL STATUS, #8420
                                5E 08 C2 00009 BEQL 4$
                                04 AE 00000000' 00 9E 0000C 2$: CMPL STATUS, #8428
                                7E 0200 5E DD 00014 BEQL 4$
                                0C 8F 3C 00016 CMPL STATUS, #2104
                                00000000V 00 03 FB 0001E BNEQ 3$
                                52 50 D0 00025 PUSHAB P.AAB
                                0E 52 E9 00028 CALLS #1, NML_RESPONSE
                                04 AE 9F 0002D BRB 1$
                                00000000G 00 02 FB 00030 3$: PUSHL STATUS
                                8F 52 D1 00039 4$: CALLS #1, LIB$STOP
                                000020E4 8F 28 13 00040 RET
                                000020EC 8F 52 D1 00042
                                00000838 8F 1F 13 00049
                                00000000' 00 0B 12 00052
                                63 01 FB 0005A
                                00000000G 00 52 DD 0005F
                                01 FB 00061
                                A2 11 00068
                                04 0006A 4$:

```

; Routine Size: 107 bytes, Routine Base: \$CODE\$ + 0024


```
237 0233 1 %SBTTL 'NML_INITIALIZE Initialization routine'
238 0234 1 ROUTINE NML_INITIALIZE =
239 0235 1
240 0236 1 ++
241 0237 1 FUNCTIONAL DESCRIPTION:
242 0238 1
243 0239 1 This is the initialization routine for the DECnet-VAX Network
244 0240 1 Management Listener. This module establishes (accepts) the logical
245 0241 1 link connection to NCP and initializes the data bases.
246 0242 1
247 0243 1 ROUTINE VALUE:
248 0244 1 COMPLETION CODES:
249 0245 1
250 0246 1 System service error is returned if initial assign for command link
251 0247 1 fails. Subsequent system service failures cause a resource error
252 0248 1 (NML$STS_RES) to be returned. If the connect data is invalid or
253 0249 1 incompatible a compatibility error (NML$STS_MVE) is returned.
254 0250 1 If the connect accept fails a network error code is returned.
255 0251 1
256 0252 1 --
257 0253 1
258 0254 2 BEGIN
259 0255 2
260 0256 2 LOCAL
261 0257 2 INI_IOSB : $IOSB, ! Connect/reject I/O status block
262 0258 2 NCB_DSC : DESCRIPTOR, ! Connect NCB descriptor
263 0259 2 STATUS; ! Temporary status
264 0260 2
265 0261 2 Determine the mode in which NML is running.
266 0262 2
267 0263 2 NML_GETMODE ();
268 0264 2
269 0265 2 Dispatch to proper initialization routine.
270 0266 2
271 0267 2 SELECTU .NML$B_MODE OF
272 0268 2 SET
273 0269 2 [NML$_CONNECT]:
274 0270 2
275 0271 2 STATUS = NML_CONNECT ();
276 0272 2
277 0273 2 [OTHERWISE]:
278 0274 2
279 0275 2 STATUS = NML$_STS_MPR;
280 0276 2
281 0277 2 TES;
282 0278 2
283 0279 2 RETURN .STATUS ! Return status code
284 0280 2
285 0281 1 END; ! End of NML_INITIALIZE
```

```
000C 00000 NML_INITIALIZE:
SE 10 C2 00002 .WORD Save R2,R3
SUBL2 #16, SP
```

```
: 0234
:
```

NML\$MAIN
V04-000

Network Management Listener main module
NML_INITIALIZE Initialization routine

B 14
15-Sep-1984 23:57:11
14-Sep-1984 12:50:12

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[NML.SRC]NMLMAIN.B32;1

Page 10
(5)

00000000V	00	00	FB 00005	CALLS	#0, NML_GETMODE	:	0263
	53	00000000'	00 9A 0000C	MOVZBL	NML\$B_MODE, R3	:	0267
	52		01 D0 00013	MOVL	#1, R2	:	
	01		53 91 00016	CMPB	R3, #1	:	0269
			09 12 00019	BNEQ	1\$	:	
			52 D4 0001B	CLRL	R2	:	
00000000V	00	00	FB 0001D	CALLS	#0, NML_CONNECT	:	0271
	03	52	E9 00024 1\$:	BLBC	R2, 2\$	:	0273
	50	0A	CE 00027	MNEGL	#10, STATUS	:	0275
		04	0002A 2\$:	RET		:	0281

; Routine Size: 43 bytes, Routine Base: \$CODE\$ + 008F

; 286 0282 1


```

288 0283 1 %SBTTL 'NML_CONNECT Connect routine'
289 0284 1 ROUTINE NML_CONNECT =
290 0285 1
291 0286 1 ++
292 0287 1 FUNCTIONAL DESCRIPTION:
293 0288 1
294 0289 1 This establishes (accepts) the logical link connection to the
295 0290 1 command process.
296 0291 1
297 0292 1 IMPLICIT OUTPUTS:
298 0293 1 NML$W_CMD_CHAN Channel number for logical link to NCP
299 0294 1
300 0295 1 ROUTINE VALUE:
301 0296 1 COMPLETION CODES:
302 0297 1
303 0298 1 System service error is returned if initial assign for command link
304 0299 1 fails. Subsequent system service failures cause a resource error
305 0300 1 (NML$STS_RES) to be returned. If the connect data is invalid or
306 0301 1 incompatible a compatibility error (NML$STS_MVE) is returned.
307 0302 1 If the connect accept fails a network error code is returned.
308 0303 1
309 0304 1 --
310 0305 1
311 0306 2 BEGIN
312 0307 2
313 0308 2 LOCAL
314 0309 2 INI_IOSB : $IOSB, ! Connect/reject I/O status block
315 0310 2 NCB_DSC : DESCRIPTOR,
316 0311 2 STATUS; ! Temporary status
317 0312 2
318 0313 2 Assign channel for link to command process
319 0314 2
320 P 0315 2 STATUS = $ASSIGN (DEVNAM = %ASCID 'NET:',
321 0316 2 CHAN = NML$W_CMD_CHAN);
322 0317 2 IF .STATUS THEN
323 0318 3 BEGIN
324 0319 3
325 0320 3 Check optional connect data for compatibility
326 0321 3
327 0322 3 STATUS = NML_VALIDLINK ();
328 0323 3 NCB_DSC [DSC$A_POINTER] = .NML$A_NCBADDRESS;
329 0324 3 IF .STATUS THEN
330 0325 4 BEGIN
331 0326 4
332 0327 4 Set up NCB to accept the connection. Add three bytes of version
333 0328 4 number to the NCB that was received. Also use the length of the
334 0329 4 received NCB.
335 0330 4
336 0331 4 CH$WCHAR A (3, NML$A_NCBDATA);
337 0332 4 IF CH$EQC (3, UPLIT BYTE (2,0,0),
338 0333 4 3, NML$A_NCBDATA) THEN
339 0334 4 CH$MOVE (3, UPLIT BYTE (2,0,0), .NML$A_NCBDATA)
340 0335 4 ELSE
341 0336 4 CH$MOVE (3, NML VERSION, .NML$A_NCBDATA);
342 0337 4 NCB_DSC [DSC$W_LENGTH] = .NML$L_CMDNCBLEN;
343 0338 4
344 0339 4 Accept connection to command process.
```

```

345      !
346      ! STATUS = $QIOW (CHAN = .NML$W_CMD_CHAN,
347      !                  FUNC = IOS_ACCESS,
348      !                  IOSB = INI_IOSB,
349      !                  P2   = NCB$DSC);
350      !
351      ! IF .STATUS THEN
352      !     STATUS = .INI_IOSB [IOS$W_STATUS]; ! Network error
353      !
354      ! ELSE
355      !     BEGIN
356      !         ! Reject connection due to incompatibility. Add one byte reject code
357      !         ! to the received NCB.
358      !         !
359      !         CH$WCHAR_A (1, NML$A_NCB$DATA);
360      !         CH$WCHAR (.STATUS / 2, .NML$A_NCB$DATA);
361      !         NCB$DSC [DSC$W_LENGTH] = .NML$C_CMD$NCB$LEN;
362      !         !
363      !         $QIOW (CHAN = .NML$W_CMD_CHAN, ! Channel
364      !                 FUNC = IOS_ACCESS OR IOSM_ABORT, ! Reject function code
365      !                 P2   = NCB$DSC); ! Reject NCB
366      !
367      !     END;
368      ! END;
369      !
370      ! RETURN .STATUS ! Return status code
371      !
372      ! END; ! End of NML_CONNECT
373
```

```

                                .PSECT $SPLITS$,NOWRT,NOEXE,2
00 00 00 3A 54 45 4E 5F 0000C P.AAE: .ASCII \ NET:\<0><0><0>
                                010E0005, 00014 P.AAD: .LONG 17694725
                                00000000, 00018 .ADDRESS P.AAE
                                00 00 02 0001C P.AAF: .BYTE 2, 0, 0
                                00 00 02 0001F P.AAG: .BYTE 2, 0, 0
                                .EXTRN SYSS$ASSIGN, SYSS$QIOW
                                .PSECT $CODE$,NOWRT,2
```

```

                                07FC 00000 NML_CONNECT:
                                .WORD Save R2,R3,R4,R5,R6,R7,R8,R9,R10 : 0284
                                5A 00000000G 00 9E 00002 MOVAB SYSS$QIOW, R10
                                59 00000000' 00 9E 00009 MOVAB P.AAD, R9
                                58 00000000' 00 9E 00010 MOVAB NML$A_NCB$DATA, R8
                                5E          10 C2 00017 SUBL2 #16, SP
                                F8          7E 7C 0001A CLRQ -(SP) : 0316
                                59 DD 0001F PUSHAB NML$W_CMD_CHAN
                                04 FB 00021 PUSHL R9
                                56          50 D0 00028 CALLS #4, SYSS$ASSIGN
                                5E          56 E9 0002B MOVL R0, STATUS
                                00          00 FB 0002E BLBC STATUS, 3$ : 0317
                                00000000V 00          CALLS #0, NML_VALIDLINK : 0322
```


		56		50	D0	00035	MOVL	R0, STATUS	:	
	04	AE	FC	A8	D0	00038	MOVL	NML\$A_NCBADDRESS, NCBDS+4	:	0323
		50		68	D0	0003D	MOVL	NML\$A_NCBDATA, R0	:	0331
		57	F4	A8	D0	00040	MOVL	NML\$A_CMDNCBLN, R7	:	0337
		6E		57	B0	00044	MOVW	R7, NCBDS	:	
		55	F8	A8	3C	00047	MOVZWL	NML\$W_CMD_CHAN, R5	:	0344
		40		56	E9	0004B	BLBC	STATUS, 4\$	:	0324
		60		03	90	0004E	MOVB	#3, (R0)	:	0331
				68	D6	00051	INCL	NML\$A_NCBDATA	:	
		54		68	D0	00053	MOVL	NML\$A_NCBDATA, R4	:	0333
	64	08	A9	03	29	00056	CMPC3	#3, P.AAF, (R4)	:	0332
				08	12	0005B	BNEQ	1\$	:	
64	18	00	0B	A9	F0	0005D	INSV	P.AAG, #0, #24, (R4)	:	0334
				06	11	00063	BRB	2\$	:	
64	18	00	EC	A9	F0	00065	INSV	NML VERSION, #0, #24, (R4)	:	0336
				7E	7C	0006B	CLRQ	-(SP)	:	0344
				7E	7C	0006D	CLRQ	-(SP)	:	
			10	AE	9F	0006F	PUSHAB	NCBDS	:	
				7E	7C	00072	CLRQ	-(SP)	:	
				7E	D4	00074	CLRL	-(SP)	:	
			28	AE	9F	00076	PUSHAB	INI_IOSB	:	
				32	DD	00079	PUSHL	#50	:	
				55	DD	0007B	PUSHL	R5	:	
				7E	D4	0007D	CLRL	-(SP)	:	
		6A		0C	FB	0007F	CALLS	#12, SYSSQIOW	:	
		56		50	D0	00082	MOVL	R0, STATUS	:	
		2C		56	E9	00085	BLBC	STATUS, 5\$	:	0345
		56	08	AE	3C	00088	MOVZWL	INI_IOSB, STATUS	:	0346
				26	11	0008C	BRB	5\$	:	0324
		60		01	90	0008E	MOVB	#1, (R0)	:	0355
				68	D6	00091	INCL	NML\$A_NCBDATA	:	
		50		68	D0	00093	MOVL	NML\$A_NCBDATA, R0	:	0356
	51	56		02	C7	00096	DIVL3	#2, STATUS, R1	:	
		60		51	90	0009A	MOVB	R1, (R0)	:	
				7E	7C	0009D	CLRQ	-(SP)	:	0361
				7E	7C	0009F	CLRQ	-(SP)	:	
			10	AE	9F	000A1	PUSHAB	NCBDS	:	
				7E	7C	000A4	CLRQ	-(SP)	:	
				7E	7C	000A6	CLRQ	-(SP)	:	
		7E	0132	8F	3C	000AB	MOVZWL	#306, -(SP)	:	
				55	DD	000AD	PUSHL	R5	:	
				7E	D4	000AF	CLRL	-(SP)	:	
		6A		0C	FB	000B1	CALLS	#12, SYSSQIOW	:	
		50		56	D0	000B4	MOVL	STATUS, R0	:	0366
				04	000B7		RET		:	0368

; Routine Size: 184 bytes, Routine Base: \$CODE\$ + 00BA

```

375 0369 1 %SBTTL 'NML_VALIDLINK Version compatibility check routine'
376 0370 1 ROUTINE NML_VALIDLINK =
377 0371 1
378 0372 1 ++
379 0373 1 FUNCTIONAL DESCRIPTION:
380 0374 1
381 0375 1 The process attempting to connect to NML must pass an NCB which, in
382 0376 1 the optional data field, contains a NICE version number. This is
383 0377 1 the version of NICE which the process expects to use in communicating
384 0378 1 with NML. This routine examines the version number to make sure it
385 0379 1 is compatible with those versions of NICE spoken by this version of NML.
386 0380 1
387 0381 1 IMPLICIT INPUTS:
388 0382 1 NML$A_NCBDATA contains the pointer to the NCB from NCP.
389 0383 1
390 0384 1 IMPLICIT OUTPUTS:
391 0385 1
392 0386 1 NML$AB_RCVBUFFER contains the received NCB.
393 0387 1 NML$SL_CMDNCBLEN contains the length of the received NCB.
394 0388 1 NML$A_NCBDATA contains the pointer to the optional NCB connect data
395 0389 1 in NML$AB_RCVBUFFER.
396 0390 1
397 0391 1 ROUTINE VALUE:
398 0392 1 COMPLETION CODES:
399 0393 1
400 0394 1 If receive Network Management version number is not greater
401 0395 1 than or equal to our own then NML$STS_MVE is returned otherwise
402 0396 1 success (NML$STS_SUC).
403 0397 1
404 0398 1 --
405 0399 1
406 0400 2 BEGIN
407 0401 2
408 0402 2 LOCAL
409 0403 2 NCP_VERSION : VECTOR [3,BYTE]; ! Command node NICE version number
410 0404 2
411 0405 2
412 0406 2 Find optional data in NCB (two bytes past '/' character)
413 0407 2
414 0408 2 NML$A_NCBDATA = CH$FIND_CH (.NML$SL_CMDNCBLEN,
415 0409 2 .NML$A_NCBADDRESS,
416 0410 2 %C'/');
417 0411 2
418 0412 2 IF NOT CH$FAIL (.NML$A_NCBDATA) THEN
419 0413 2 NML$A_NCBDATA = .NML$A_NCBDATA + 3 ! Skip '/' and two bytes
420 0414 2 ELSE
421 0415 2 RETURN NML$STS_MPR; ! Return error
422 0416 2
423 0417 2 Compare received version with our own. Value greater than or equal
424 0418 2 is successful (Phase III). Less than three bytes of optional data or
425 0419 2 value less than our own but not version 2.0.0 or 3.0.0 fails.
426 0420 2
427 0421 2 IF CH$RCHAR (.NML$A_NCBDATA) EQLU 0 THEN
428 0422 2
429 0423 2 If no version number in the NCB, assume Phase II, NICE V1.3.0.
430 0424 2
431 0425 2 BEGIN
```



```

: 432      0426 3      CH$MOVE (3, UPLIT BYTE(1,3,0), NCP_VERSION);
: 433      0427 3      END
: 434      0428 2      ELSE
: 435      0429 3      BEGIN
: 436      0430 3      IF CH$RCHAR (.NML$A_NCBDATA) EQLU 3 THEN
: 437      0431 3      CH$MOVE (3, .NML$A_NCBDATA+1, NCP_VERSION)
: 438      0432 3      ELSE
: 439      0433 3      RETURN NML$_STS_MVE;
: 440      0434 2      END;
: 441      0435 2      IF NML$INITIALIZE (NCP_VERSION) THEN
: 442      0436 2      RETURN NML$_STS_SUC
: 443      0437 2      ELSE
: 444      0438 2      RETURN NML$_STS_MVE;
: 445      0439 1      END;
                                ! End of NML_VALIDLINK
```

.PSECT \$SPLITS,NOWRT,NOEXE,2

00 03 01 00022 P.AAH: .BYTE 1, 3, 0 ;

.PSECT \$CODE\$,NOWRT,2

				0004 00000 NML_VALIDLINK:			
		52	00000000'	00 9E 00002	.WORD	Save R2	0370
		5E		04 C2 00009	MOVAB	NML\$A_NCBDATA, R2	
		50	FC	A2 D0 0000C	SUBL2	#4, SP	
	60	A2		2F 3A 00010	MOVL	NML\$A_NCBADDRESS, R0	0409
				02 12 00015	LOCC	#47, NML\$A_CMDNCBLEN, (R0)	0408
				51 D4 00017	BNEQ	1\$	
					CLRL	R1	
		62		51 D0 00019	1\$: MOVL	R1, NML\$A_NCBDATA	
				05 13 0001C	BEQL	2\$	0412
		62		03 C0 0001E	ADDL2	#3, NML\$A_NCBDATA	0413
				04 11 00021	BRB	3\$	
		50		0A CE 00023	2\$: MNEGL	#10, R0	0415
				04 00026	RET		
		50		62 D0 00027	3\$: MOVL	NML\$A_NCBDATA, R0	0421
				60 95 0002A	TSTB	(R0)	
				0B 12 0002C	BNEQ	4\$	
6E	18		00 00000000'	00 F0 0002E	INSV	P.AAH, #0, #24, NCP_VERSION	0426
				0B 11 00037	BRB	5\$	0421
		03		60 91 00039	4\$: CMPB	(R0), #3	0430
				16 12 0003C	BNEQ	6\$	
6E	18		00 01	A0 F0 0003E	INSV	1(R0), #0, #24, NCP_VERSION	0431
				5E DD 00044	5\$: PUSHL	SP	0435
		00000000G	00	01 FB 00046	CALLS	#1, NML\$INITIALIZE	
			04	50 E9 0004D	BLBC	R0, 6\$	
			50	01 D0 00050	MOVL	#1, R0	0438
				04 00053	RET		
			50	0E CE 00054	6\$: MNEGL	#14, R0	
				04 00057	RET		0439

; Routine Size: 88 bytes, Routine Base: \$CODE\$ + 0172

NML\$MAIN
V04-000

Network Management Listener main module
NML_VALIDLINK Version compatibility check routi

H 14
15-Sep-1984 23:57:11
14-Sep-1984 12:50:12

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[NML.SRC]NMLMAIN.B32;1 Page 16
(7)

NM
VO


```

447 0440 1 %SBTTL 'NML_GETMODE Determine the mode in which NML is running'
448 0441 1 ROUTINE NML_GETMODE : NOVALUE =
449 0442 1
450 0443 1 ++
451 0444 1 FUNCTIONAL DESCRIPTION:
452 0445 1 This routine determines the mode in which NML is running by
453 0446 1 determining how the image was activated. NML runs one of in two
454 0447 1 modes:
455 0448 1 Communicating with NCP on a remote node via a logical link,
456 0449 1 processing NICE commands.
457 0450 1 In autoservice mode - doing down line loads and loopback
458 0451 1 functions requested by other nodes using the MOP
459 0452 1 protocol.
460 0453 1
461 0454 1 FORMAL PARAMETERS:
462 0455 1
463 0456 1 NONE
464 0457 1
465 0458 1 IMPLICIT INPUTS:
466 0459 1
467 0460 1 NONE
468 0461 1
469 0462 1 IMPLICIT OUTPUTS:
470 0463 1
471 0464 1 NML$B_MODE contains the mode in which NML is running.
472 0465 1 (it may be unchanged if we cannot determine the mode)
473 0466 1
474 0467 1 ROUTINE VALUE:
475 0468 1 COMPLETION CODES:
476 0469 1
477 0470 1 NONE
478 0471 1
479 0472 1 SIDE EFFECTS:
480 0473 1
481 0474 1 NONE
482 0475 1
483 0476 1 --
484 0477 1
485 0478 2 BEGIN
486 0479 2
487 0480 2 LOCAL
488 0481 2 STATUS; ! Temporary status
489 0482 2
490 0483 2
491 0484 2 Get optional connect data or service circuit name by translating SYS$NET
492 0485 2
493 P 0486 2 STATUS = $TRNLNM (ATTR = UPLIT (LNMSM CASE BLIND),
494 P 0487 2 TABNAM = %ASCID 'LNMS$PROCESS_TABLE',
495 P 0488 2 LOGNAM = %ASCID 'SYS$NET',
496 P 0489 2 ITMLST = UPLIT (WORD (nml$k_rcvbflen),
497 P 0490 2 WORD (lnm$string),
498 P 0491 2 LONG (nml$ab_rcvbuffer),
499 P 0492 2 LONG (nml$l_cmdncblen),
500 P 0493 2 LONG (0))
501 0494 2 );
502 0495 2 IF .STATUS EQLU SSS_NORMAL
503 0496 2 THEN
```


NML\$MAIN
V04-000

Network Management Listener main module
NML_GETMODE Determine the mode in which NML is

J 14
15-Sep-1984 23:57:11
14-Sep-1984 12:50:12

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[NML.SRC]NMLMAIN.B32;1 Page 18
(8)

```

: 504      0497      3      BEGIN
: 505      0498      3
: 506      0499      3      Since SYS$NET translates, NML was activated by a connect request
: 507      0500      3      from an NCP on a remote node.
: 508      0501      3
: 509      0502      3      NML$B_MODE = NML$_CONNECT;
: 510      0503      3      NML$_NCBADDRESS = NML$AB_RCVBUFFER; ! Initialize NCB pointer
: 511      0504      2      END;
: 512      0505      2
: 513      0506      1      END;
                                ! End of NML_GETMODE
```

```

                                .PSECT $SPLIT$,NOWRT,NOEXE,2
                                .BLKB 3
                                .LONG 33554432
                                .ASCII \LNMS$PROCESS_TABLE\<0><0><0>
                                .LONG 17694737
                                .ADDRESS P.AAK
                                .ASCII \SYS$NET\<0>
                                .LONG 17694727
                                .ADDRESS P.AAM
                                .WORD 512
                                .WORD 2
                                .ADDRESS NML$AB_RCVBUFFER
                                .ADDRESS NML$_CMDNCBLEN
                                .LONG 0
                                .EXTRN SYS$TRNLNM
                                .PSECT $CODE$,NOWRT,2
                                000C 00000 NML_GETMODE:
                                .WORD Save R2,R3
                                MOVAB NML$B_MODE, R3
                                MOVAB P.AAN, R2
                                PUSHL R2
                                CLRL -(SP)
                                PUSHAB P.AAL
                                PUSHAB P.AAJ
                                PUSHAB P.AAI
                                CALLS #5, SYS$TRNLNM
                                CMPL STATUS, #1
                                BNEQ 1$
                                MOVB #1, NML$B_MODE
                                MOVAB NML$AB_RCVBUFFER, NML$_NCBADDRESS
                                RET
                                04 00031 1$:
                                53 00000000' 00 9E 00002
                                52 00000000' 00 9E 00009
                                52 DD 00010
                                7E D4 00012
                                F8 A2 9F 00014
                                E8 A2 9F 00017
                                D0 A2 9F 0001A
                                00000000G 00 05 FB 0001D
                                01 50 D1 00024
                                63 08 12 00027
                                OC A3 01 90 00029
                                A3 1C A3 9E 0002C
                                04 00031 1$:
                                00025
                                00028 P.AAI:
                                0002C P.AAK:
                                0003B
                                00040 P.AAJ:
                                00044
                                00048 P.AAM:
                                00050 P.AAL:
                                00054
                                00058 P.AAN:
                                0005A
                                0005C
                                00060
                                00064
```

; Routine Size: 50 bytes, Routine Base: \$CODE\$ + 01CA


```

515 0507 1 %SBTTL 'NML_RECEIVE Network receive routine'
516 0508 1 ROUTINE NML_RECEIVE (BUFADR, BUFLN, RCVLEN) =
517 0509 1
518 0510 1 ++
519 0511 1 FUNCTIONAL DESCRIPTION:
520 0512 1
521 0513 1 This routine receives NICE protocol messages over a logical link
522 0514 1 from a command process (generally NCP).
523 0515 1
524 0516 1 FORMAL PARAMETERS:
525 0517 1
526 0518 1 BUFADR Address of the receive buffer.
527 0519 1 BUFLN Length of the receive buffer.
528 0520 1 RCVLEN Actual length of the received data if successful.
529 0521 1
530 0522 1 IMPLICIT INPUTS:
531 0523 1
532 0524 1 NML$W_CMD_CHAN Channel assigned to the command process link.
533 0525 1
534 0526 1 IMPLICIT OUTPUTS:
535 0527 1
536 0528 1 The buffer pointed to by BUFADR contains the received data.
537 0529 1
538 0530 1 ROUTINE VALUE:
539 0531 1 COMPLETION CODES:
540 0532 1
541 0533 1 System service completion status is returned if receive fails
542 0534 1 indicating that the link has been terminated for some reason.
543 0535 1 Otherwise success is returned.
544 0536 1
545 0537 1 SIDE EFFECTS:
546 0538 1
547 0539 1 NONE
548 0540 1
549 0541 1 --
550 0542 1
551 0543 2 BEGIN
552 0544 2
553 0545 2 LOCAL
554 0546 2 RCV_IOSB : $IOSB, ! Receive I/O status block
555 0547 2 STATUS; ! Temporary status
556 0548 2
557 0549 2 Receive a NICE message
558 0550 2
559 0551 2 STATUS = $QIOW (FUNC = IOS$ READVBLK, ! Function
560 0552 2 CHAN = .NML$W_CMD_CHAN, ! Channel
561 0553 2 IOSB = RCV_IOSB, ! I/O status block
562 0554 2 P1 = .BUFADR, ! Buffer address
563 0555 2 P2 = .BUFLN); ! Buffer length
564 0556 2
565 0557 2 If system service completed successfully get network status
566 0558 2
567 0559 2 IF .STATUS
568 0560 2 THEN
569 0561 2 STATUS = .RCV_IOSB [IOS$W_STATUS];
570 0562 2
571 0563 2 If network status is success then get received message length
```

NML\$MAIN
V04-000

Network Management Listener main module
NML_RECEIVE Network receive routine

L 14
15-Sep-1984 23:57:11
14-Sep-1984 12:50:12

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[NML.SRC]NMLMAIN.B32;1
Page 20
(9)

```
: 572      0564  2  !  
: 573      0565  2  !  
: 574      0566  2  !  
: 575      0567  2  !  
: 576      0568  2  !  
: 577      0569  2  !  
: 578      0570  2  !  
: 579      0571  1  !  
  
      IF .STATUS  
      THEN  
          .RCVLEN = .RCV_IOSB [IOS$W_COUNT];      ! Return received data length  
  
      RETURN .STATUS      ! Return status code  
  
      END;      ! End of NML$RECEIVE
```

```
0000 00000 NML_RECEIVE:  
      5E      08 C2 00002      .WORD      Save nothing      : 0508  
      7E      7E 7C 00005      SUBL2      #8, SP  
      7E      7E 7C 00007      CLRQ      -(SP)      : 0555  
      7E      04 AC 7D 00009      CLRQ      -(SP)  
      20      7E 7C 0000D      MOVQ      BUFADR, -(SP)  
      31      AE 9F 0000F      CLRQ      -(SP)  
      7E 00000000' 00 3C 00014      PUSHAB  RCV_IOSB  
      7E      D4 0001B      PUSHL      #49  
      00000000G 00 0C FB 0001D      MOVZWL  NML$W_CMD_CHAN, -(SP)  
      08      50 E9 00024      CLRL      -(SP)  
      50      6E 3C 00027      CALLS     #12, SYSSQIOW  
      05      50 E9 0002A      BLBC      STATUS, 1$      : 0559  
      0C      BC      02 AE 3C 0002D      MOVZWL  RCV_IOSB, STATUS      : 0561  
      04 00032 1$:      BLBC      STATUS, 1$      : 0565  
      RET      MOVZWL  RCV_IOSB+2, @RCVLEN      : 0567  
      : 0571
```

; Routine Size: 51 bytes, Routine Base: \$CODE\$ + 01FC

; 580 0572 1


```

582 0573 1 %SBTTL 'NML_RESPONSE Network response routine'
583 0574 1 ROUTINE NML_RESPONSE (BUFDISC): NOVALUE =
584 0575 1
585 0576 1 ++
586 0577 1
587 0578 1 This routine is called to process a NICE response as a result
588 0579 1 of processing a NICE message. This routine may be called as
589 0580 1 many times as necessary during the processing of a single NICE
590 0581 1 message.
591 0582 1
592 0583 1 Inputs:
593 0584 1
594 0585 1 bufdesc Address of descriptor of message to be transmitted.
595 0586 1
596 0587 1 nml$w_cmd_chan Channel assigned to the command process link.
597 0588 1
598 0589 1 Outputs:
599 0590 1
600 0591 1 Returns success. Errors are signalled.
601 0592 1 --
602 0593 1
603 0594 2 BEGIN
604 0595 2
605 0596 2 MAP
606 0597 2 bufdesc: REF BLOCK [,BYTE]; ! Address of descriptor
607 0598 2
608 0599 2 LOCAL
609 0600 2 snd_iosb : $IOSB, ! Send I/O status block
610 0601 2 status; ! Temporary status
611 0602 2
612 0603 2
613 0604 2 Send the NICE response message
614 0605 2
615 0606 2
616 P 0607 2 status = $QIOW (FUNC = IOS_WRITEVBLK,
617 PP 0608 2 CHAN = .nml$w_cmd_chan,
618 PP 0609 2 IOSB = snd_iosb,
619 P 0610 2 P1 = .bufdesc [dsc$a_pointer],
620 0611 2 P2 = .bufdesc [dsc$w_length]);
621 0612 2
622 0613 2 If system service was successful then get network status
623 0614 2
624 0615 2
625 0616 2 IF .STATUS
626 0617 2 THEN
627 0618 2 STATUS = .SND_IOSB [IOS$W_STATUS];
628 0619 2
629 0620 2
630 0621 2 If status is bad then signal the error.
631 0622 2
632 0623 2
633 0624 2 IF NOT .STATUS
634 0625 2 THEN
635 0626 2 SIGNAL_STOP (.STATUS);
636 0627 2
637 0628 1 END;
```

		0000 00000 NML_RESPONSE:					
5E		08	C2	00002	.WORD	Save nothing	: 0574
		7E	7C	00005	SUBL2	#8, SP	: 0611
		7E	7C	00007	CLRQ	-(SP)	
50	04	AC	D0	00009	CLRQ	-(SP)	
7E		60	3C	0000D	MOVL	BUFDESC, R0	
	04	A0	DD	00010	MOVZWL	(R0), -(SP)	
		7E	7C	00013	PUSHL	4(R0)	
	20	AE	9F	00015	CLRQ	-(SP)	
		30	DD	00018	PUSHAB	SND_IOSB	
7E	00000000'	00	3C	0001A	PUSHL	#48	
		7E	D4	00021	MOVZWL	NML\$W_CMD_CHAN, -(SP)	
00000000G	00	0C	FB	00023	CLRL	-(SP)	
	06	50	E9	0002A	CALLS	#12, SYS\$QIOW	: 0616
	50	6E	3C	0002D	BLBC	STATUS, 1\$	: 0618
	09	50	E8	00030	MOVZWL	SND_IOSB, STATUS	: 0624
		50	DD	00033	BLBS	STATUS, 2\$	: 0626
00000000G	00	01	FB	00035	PUSHL	STATUS	: 0628
		04	0003C	2\$:	CALLS	#1, LIB\$STOP	
					RET		

; Routine Size: 61 bytes, Routine Base: \$CODE\$ + 022F

.EXTRN LIB\$STOP

PSECT SUMMARY

Name	Bytes	Attributes					
\$PLITS	104	NOVEC,NOWRT,	RD	NOEXE,NOSHR,	LCL,	REL,	CON,NOPIC,ALIGN(2)
\$OWNS	540	NOVEC, WRT,	RD	NOEXE,NOSHR,	LCL,	REL,	CON,NOPIC,ALIGN(2)
\$CODES	620	NOVEC,NOWRT,	RD	EXE,NOSHR,	LCL,	REL,	CON,NOPIC,ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
-\$255\$DUA28:[NML.OBJ]NMLLIB.L32;1	341	15	4	27	00:00.1
-\$255\$DUA28:[SHRLIB]NMLIBRY.L32;1	887	1	0	47	00:00.2
-\$255\$DUA28:[SHRLIB]NET.L32;1	1279	0	0	63	00:00.3
-\$255\$DUA28:[SYSLIB]STARLET.L32;1	9776	18	0	581	00:03.3

COMMAND QUALIFIERS

```

:
:      BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:NMLMAIN/OBJ=OBJ$:NMLMAIN MSRC$:NMLMAIN/UPDATE=(ENHS:NMLMAIN)
:
: Size:          620 code + 644 data bytes
: Run Time:      00:16.8
: Elapsed Time:  00:41.9
: Lines/CPU Min: 2255
: Lexemes/CPU-Min: 14233
: Memory Used:   101 pages
: Compilation Complete

```


0284 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

