

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	

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

[illegible]

NC
VC

```
0001 0 %TITLE 'Network I/O Routines'
0002 0 MODULE NCPNETIO (IDENT = 'V04-000',
0003 0 ADDRESSING_MODE(EXTERNAL=GENERAL),
0004 0 ADDRESSING_MODE(NONEXTERNAL=GENERAL)) =
0005 1 BEGIN
0006 1
0007 1
0008 1 *****
0009 1 *
0010 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0011 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0012 1 * ALL RIGHTS RESERVED.
0013 1 *
0014 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0015 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0016 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0017 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0018 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0019 1 * TRANSFERRED.
0020 1 *
0021 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0022 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0023 1 * CORPORATION.
0024 1 *
0025 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0026 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0027 1 *
0028 1 *
0029 1 *****
0030 1
0031 1
0032 1 **
0033 1 FACILITY:      Network Control Program (NCP)
0034 1
0035 1 ABSTRACT:
0036 1
0037 1     This module contains the routines which establish and communicate
0038 1     over a logical link to NML on the executor node.
0039 1
0040 1 ENVIRONMENT:   VAX/VMS Operating System
0041 1
0042 1 AUTHOR:       Darrell Duffy   , CREATION DATE: 30-October-1979
0043 1
0044 1 MODIFIED BY:
0045 1
0046 1     V03-005 PRD0097      Paul R. DeStefano      19-Apr-1984
0047 1     Disable defaulting of area to 1.n if area is not
0048 1     specified (defeats PRD0087).
0049 1
0050 1     V03-004 PRD0094      Paul R. DeStefano      09-Apr-1984
0051 1     Modified NCP$READRSP and NCP$CONERR routines to
0052 1     interpret the error detail when the error status
0053 1     is NMASC_STS_OPE (operation failure). Additional
0054 1     text is now appended to the original error text
0055 1     in the nice message based on the detail.
0056 1
0057 1     V03-003 PRD0087      Paul R. DeStefano      27-Mar-1984
```

58	0058	1	Make SET EXEC NODE n default to 1.n if area is not
59	0059	1	specified.
60	0060	1	
61	0061	1	V03-002 RPG0002 Bob Grosso 20-Apr-1983
62	0062	1	Add NCP\$CONERR for the CONNECT routine to use to
63	0063	1	process errors.
64	0064	1	
65	0065	1	V03-001 RPG0001 Bob Grosso 16-Mar-1983
66	0066	1	Update version number checking for version IV.
67	0067	1	
68	0068	1	V009 TMH0009 Tim Halvorsen 11-Jan-1982
69	0069	1	Save version number of NML server in LCB.
70	0070	1	Make NCP\$OPENLINK a global routine.
71	0071	1	
72	0072	1	V008 TMH0008 Tim Halvorsen 15-Dec-1981
73	0073	1	Print detail messages for FCO NICE errors.
74	0074	1	
75	0075	1	V007 TMH0007 Tim Halvorsen 22-Oct-1981
76	0076	1	Fix the spelling on some messages.
77	0077	1	
78	0078	1	V006 LMK0006 Len Kowell 19-Sep-1981
79	0079	1	Change version checking to allow current or greater and V2.0.
80	0080	1	
81	0081	1	V005 TMH0005 Tim Halvorsen 11-Aug-1981
82	0082	1	Use different detail text table if looking up system-
83	0083	1	specific entity number. When formatting a parameter
84	0084	1	detail, use the signed entity in NCP\$GL_ENTITY rather
85	0085	1	than the option byte, since it doesn't tell whether
86	0086	1	its a system-specific entity or not. Only supply a
87	0087	1	comma following an NML response message if there is
88	0088	1	a non-blank detail following it.
89	0089	1	
90	0090	1	V004 TMH0004 Tim Halvorsen 10-Jul-1981
91	0091	1	Change all non-local references to use general addressing.
92	0092	1	Use new callable NML whenever we are communicating with the
93	0093	1	local node without any access control string.
94	0094	1	
95	0095	1	V003 TMH0003 Tim Halvorsen 06-Jul-1981
96	0096	1	Remove version # checks on NML connect to allow communication
97	0097	1	between a 2.2 NCP and a 2.0 NML, which normally should not be
98	0098	1	allowed, but will be for compatibility after 2.2 release.
99	0099	1	
100	0100	1	V02-002 LMK0001 Len Kowell 29-Sep-1980
101	0101	1	Change \$DELMBX call to \$DASSGN call.
102	0102	1	--

```

: 104      0103 1 %SBTTL 'Definitions'
: 105      0104 1
: 106      0105 1
: 107      0106 1 : TABLE OF CONTENTS:
: 108      0107 1 :
: 109      0108 1
: 110      0109 1 FORWARD ROUTINE
: 111      0110 1      NCP$BLDLCB : NOVALUE,
: 112      0111 1      NCP$OPENLINK : NOVALUE,
: 113      0112 1      NCP$SIGNETERR : NOVALUE,
: 114      0113 1      NCP$CLOSELINK : NOVALUE,
: 115      0114 1      NCP$SENDMSG : NOVALUE,
: 116      0115 1      STORE_RESPONSE : NOVALUE,
: 117      0116 1      NCP$READRSP,
: 118      0117 1      NCP$TABLESEARCH
: 119      0118 1      :
: 120      0119 1
: 121      0120 1 : INCLUDE FILES:
: 122      0121 1 :
: 123      0122 1 :
: 124      0123 1 :
: 125      0124 1      LIBRARY 'SYSS$LIBRARY:STARLET.L32';
: 126      0125 1      LIBRARY 'OBJ$:NMALIBRY.L32';
: 127      0126 1      LIBRARY 'OBJ$:NCPLIBRY.L32';
: 128      0127 1
: 129      0128 1 :
: 130      0129 1 : MACROS:
: 131      0130 1 :
: 132      0131 1 :
: 133      0132 1 :
: 134      0133 1 : EQUATED SYMBOLS:
: 135      0134 1 :
: 136      0135 1 :
: 137      0136 1 :
: 138      0137 1 : Trailing portion of the Network Connect Block (NCB)
: 139      0138 1 :
: 140      0139 1 BIND
: 141      0140 1
: 142      P 0141 1      NCP$Q_OBJSPEC = ASCID ('::'19=/' , %CHAR(0,0),
: 143      P 0142 1      NCP$Q_OBJSPEC = ASCID ('::'0=NML/' , %CHAR(0,0),
: 144      P 0143 1      %CHAR(3, NCP$C_VRS, NCP$C_ECO, NCP$C_UECO),
: 145      P 0144 1      )
: 146      0145 2
: 147      0146 1 :
: 148      0147 1
```

```

: 150      0148 1
: 151      0149 1
: 152      0150 1  : OWN STORAGE:
: 153      0151 1
: 154      0152 1
: 155      0153 1
: 156      0154 1  : Mailbox and Response buffers
: 157      0155 1
: 158      0156 1
: 159      0157 1 GLOBAL
: 160      0158 1      NCP$GT_MBXBFR : VECTOR [NCP$C_MBXSIZE, BYTE],
: 161      0159 1      NCP$GT_RSPBFR : VECTOR [NCP$C_RSPSIZE, BYTE]
: 162      0160 1      ;
: 163      0161 1
: 164      0162 1
: 165      0163 1  : Data to maintain the link control blocks for the executor
: 166      0164 1
: 167      0165 1
: 168      0166 1 GLOBAL
: 169      0167 1      NCP$GT_EXECLCB : BBLOCK [LCB$C_SIZE],
: 170      0168 1      NCP$GT_TELLCLCB : BBLOCK [LCB$C_SIZE],
: 171      0169 1
: 172      0170 1      NCP$GL_OLDLCB,
: 173      0171 1      NCP$GL_EXELCB
: 174      0172 1      ;
: 175      0173 1
: 176      0174 1 OWN
: 177      0175 1      NML_RESP_QUEUE: VECTOR [2]      ! Local NML response queue header
: 178      0176 1      INITIAL(NML_RESP_QUEUE,NML_RESP_QUEUE);
: 179      0177 1
: 180      0178 1
: 181      0179 1  : EXTERNAL REFERENCES:
: 182      0180 1
: 183      0181 1
: 184      0182 1 EXTERNAL
: 185      0183 1      NCP$GL_FNC_CODE,      ! Function code for command message
: 186      0184 1      NCP$GL_ENTITY;      ! Entity number for command message
: 187      0185 1
: 188      0186 1 EXTERNAL ROUTINE
: 189      0187 1      NML$INITIALIZE: NOVALUE,      ! Initialize NML sharable image
: 190      0188 1      NML$PROCESS_NICE: NOVALUE,      ! Process a single NICE message
: 191      0189 1      NML$TERMINATE: NOVALUE,      ! Terminate NML sharable image
: 192      0190 1      LIB$GET_VM,      ! Allocate dynamic memory
: 193      0191 1      LIB$FREE_VM,      ! Deallocate dynamic memory
: 194      0192 1      NCP$FORMATPARM : NOVALUE;      ! Format a parameter as text
```



```

196 0193 1 %SBTTL 'ACT$VRB_TELL Process TELL Verb'
197 0194 1 GLOBAL ROUTINE ACT$VRB_TELL = !
198 0195 1
199 0196 1 !++
200 0197 1 FUNCTIONAL DESCRIPTION:
201 0198 1
202 0199 1 Action routine to setup an executor node for one command.
203 0200 1 Current executor LCB is saved and a new one is setup.
204 0201 1 A link is opened to the new executor node.
205 0202 1
206 0203 1 FORMAL PARAMETERS:
207 0204 1
208 0205 1 NONE
209 0206 1
210 0207 1 IMPLICIT INPUTS:
211 0208 1
212 0209 1 NCP$GL_OLDLCB Save the current executor lcb
213 0210 1 NCP$GL_EXELCB The current executor lcb
214 0211 1 NCP$GT_TELL_LCB LCB to use for tell
215 0212 1
216 0213 1 IMPLICIT OUTPUTS:
217 0214 1
218 0215 1 NCP$GT_TELL_LCB Link opened
219 0216 1
220 0217 1 ROUTINE VALUE:
221 0218 1 COMPLETION CODES:
222 0219 1
223 0220 1 Success or error signaled
224 0221 1
225 0222 1 SIDE EFFECTS:
226 0223 1
227 0224 1 NONE
228 0225 1
229 0226 1 --
230 0227 1
231 0228 2 BEGIN
232 0229 2
233 0230 2 NCP$GL_OLDLCB = .NCP$GL_EXELCB; ! Save the current executor
234 0231 2 NCP$GL_EXELCB = NCP$GT_TELL_LCB; ! Set the new one
235 0232 2 NCP$BLD_LCB (.NCP$GL_EXELCB); ! Build the link control block
236 0233 2 NCP$OPENLINK (.NCP$GL_EXELCB); ! Open the link
237 0234 2 RETURN SUCCESS ! Always succeed for action
238 0235 2
239 0236 1 END;

```

```

.TITLE NCPNETIO Network I/O Routines
.IDENT \V04-000\

.PSECT $SPLITS$,NOWRT,NOEXE,2

.P.AAB: .ASCII \::'19=/'\<0><0><3><4><0><0>
.P.AAA: .ASCII \
.LONG 27
.ADDRESS P.AAB

.PSECT $OWNS$,NOEXE,2

```

```

00 22 00 00 04 03 00 00 2F 3D 39 31 22 3A 3A 00000 P.AAB:
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 00000
0000001B 0001C P.AAA:
00000000 00020

```

```

00000000' 00000000' 00000 NML_RESP_QUEUE:
                                .ADDRESS NML_RESP_QUEUE, NML_RESP_QUEUE
                                .PSECT $GLOBAL$,NOEXE,2

00000 NCP$GT_MBXBFR::
                                .BLKB 40
00028 NCP$GT_RSPBFR::
                                .BLKB 1000
00410 NCP$GT_EXECLCB::
                                .BLKB 118
00486 NCP$GT_TELL_LCB::
                                .BLKB 2
00488 NCP$GT_TELL_LCB::
                                .BLKB 118
004FE NCP$GL_OLDLCB::
                                .BLKB 2
00500 NCP$GL_EXELCB::
                                .BLKB 4
00504 NCP$GL_EXELCB::
                                .BLKB 4

NCP$Q_OBJSPEC= P.AAA
                                .EXTRN NCP$GL_FNC_CODE
                                .EXTRN NCP$GL_ENTITY, NML$INITIALIZE
                                .EXTRN NML$PROCESS_NICE
                                .EXTRN NML$TERMINATE, LIB$GET_VM
                                .EXTRN LIB$FREE_VM, NCP$FORMATPARM

                                .PSECT $CODE$,NOWRT,2

                                .ENTRY ACT$VRB_TELL, Save R2
MOVAB NCP$GL_EXELCB, R2
MOVL NCP$GL_EXELCB, NCP$GL_OLDLCB
MOVAB NCP$GT_TELL_LCB, NCP$GL_EXELCB
PUSHL NCP$GL_EXELCB
CALLS #1, NCP$BLDLCB
PUSHL NCP$GL_EXELCB
CALLS #1, NCP$OPENLINK
MOVL #1, R0
RET

```

FC
52
00000000'
00
9E
00002
62
D0
00009
A2
9E
0000D
62
DD
00011
01
FB
00013
62
DD
0001A
01
FB
0001C
50
D0
00023
04
00026

0004 00000
0002 00002
0009 00009
000D 0000D
0011 00011
0013 00013
001A 0001A
001C 0001C
0023 00023
0026 00026

: 0194
: 0230
: 0231
: 0232
: 0233
: 0234
: 0236

; Routine Size: 39 bytes, Routine Base: \$CODE\$ + 0000


```
241 0237 1
242 0238 1 %SBTTL 'NCP$UNTELL Remove the TELL Executor Node'
243 0239 1 GLOBAL ROUTINE NCP$UNTELL :NOVALUE = !
244 0240 1
245 0241 1 ++
246 0242 1 FUNCTIONAL DESCRIPTION:
247 0243 1
248 0244 1 If the last command had a TELL prefix, the link to the temporary
249 0245 1 executor is broken and the previous executor node is restored.
250 0246 1
251 0247 1 FORMAL PARAMETERS:
252 0248 1
253 0249 1 NONE
254 0250 1
255 0251 1 IMPLICIT INPUTS:
256 0252 1
257 0253 1 NCP$GL_OLDLCB Pointer to previous executor LCB
258 0254 1 NCP$GL_EXELCB Tell executor LCB
259 0255 1
260 0256 1 IMPLICIT OUTPUTS:
261 0257 1
262 0258 1 NONE
263 0259 1
264 0260 1 ROUTINE VALUE:
265 0261 1 COMPLETION CODES:
266 0262 1
267 0263 1 NONE
268 0264 1
269 0265 1 SIDE EFFECTS:
270 0266 1
271 0267 1 NONE
272 0268 1
273 0269 1 --
274 0270 1
275 0271 2 BEGIN
276 0272 2
277 0273 2 IF .NCP$GL_OLDLCB NEQ 0 ! Is there a TELL outstanding?
278 0274 2 THEN
279 0275 3 BEGIN
280 0276 3 NCP$CLOSELINK (.NCP$GL_EXELCB); ! Close the link to William TELL
281 0277 3 NCP$GL_EXELCB = .NCP$GL_OLDLCB; ! Restore the old link
282 0278 3 NCP$GL_OLDLCB = 0 ! There is no William TELL now
283 0279 3 END
284 0280 2
285 0281 2 RETURN
286 0282 2
287 0283 1 END;
```

```
52 00000000' 0004 00000
00 9E 00002
62 D5 00009
10 13 0000B
04 A2 DD 0000D
```

```
.ENTRY NCP$UNTELL, Save R2
MOVAB NCP$GL_OLDLCB, R2
TSTL NCP$GL_OLDLCB
BEQL 1$
PUSHL NCP$GL_EXELCB
```

```
: 0239
:
: 0273
:
: 0276
```

NCPNETIO
V04-000

Network I/O Routines
NCP\$UNTELL Remove the TELL Executor Node

D 16
15-Sep-1984 23:46:44
14-Sep-1984 12:48:14

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[NCP.SRC]NCPNETIO.B32;1 Page 8
(5)

00000000V 00
04 A2

01 FB 00010
62 D0 00017
62 D4 0001B
04 0001D 1\$:

CALLS #1, NCP\$CLOSELINK
MOVL NCP\$GL_OLDLCB, NCP\$GL_EXELCB
CLRL NCP\$GL_OLDLCB
RET

:
: 0277
: 0278
: 0283

; Routine Size: 30 bytes, Routine Base: \$CODE\$ + 0027

```
289 0284 1 %SBTTL 'ACT$VRB_SETEXEC Establish the Executor Node'
290 0285 1 GLOBAL ROUTINE ACT$VRB_SETEXEC =
291 0286 1
292 0287 1 ++
293 0288 1 FUNCTIONAL DESCRIPTION:
294 0289 1
295 0290 1 This is an action routine to establish a link to an executor node.
296 0291 1 Any previous link to an executor is broken.
297 0292 1 The LCB is built from data left by the parse and a link is
298 0293 1 opened.
299 0294 1
300 0295 1 FORMAL PARAMETERS:
301 0296 1
302 0297 1 NONE
303 0298 1
304 0299 1 IMPLICIT INPUTS:
305 0300 1
306 0301 1 NCP$GL_OLDLCB Pointer to the lcb for the exec if tell active
307 0302 1 NCP$GL_EXELCB Pointer to the lcb for the exec
308 0303 1 NCP$GT_EXECLCB LCB to be used by the exec
309 0304 1
310 0305 1 IMPLICIT OUTPUTS:
311 0306 1
312 0307 1 NONE
313 0308 1
314 0309 1 ROUTINE VALUE:
315 0310 1 COMPLETION CODES:
316 0311 1
317 0312 1 Success or an error signaled by called routine
318 0313 1
319 0314 1 SIDE EFFECTS:
320 0315 1
321 0316 1 NONE
322 0317 1
323 0318 1 --
324 0319 1
325 0320 2 BEGIN
326 0321 2
327 0322 2 NCP$GL_OLDLCB = 0;
328 0323 2 NCP$GL_EXELCB = NCP$GT_EXECLCB;
329 0324 2 NCP$CLOSELINK (.NCP$GL_EXELCB);
330 0325 2 NCP$BLDLCB (.NCP$GL_EXELCB);
331 0326 2 NCP$OPENLINK (.NCP$GL_EXELCB);
332 0327 2 RETURN SUCCESS
333 0328 2
334 0329 1 END;
```

```
0004 00000
52 00000000' 00 9E 00002
FC A2 D4 00009
62 FF0C C2 9E 0000C
00000000V 00 62 DD 00011
01 FB 00013
```

```
.ENTRY ACT$VRB_SETEXEC, Save R2 : 0285
MOVAB NCP$GL_EXELCB, R2 :
CLRL NCP$GL_OLDLCB : 0322
MOVAB NCP$GT_EXECLCB, NCP$GL_EXELCB : 0323
PUSHL NCP$GL_EXELCB : 0324
CALLS #1, NCP$CLOSELINK :
```

NCPNET10
V04-000

Network I/O Routines
ACTSVRB_SETEXEC Establish the Executor Node

F 16
15-Sep-1984 23:46:44
14-Sep-1984 12:48:14

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[NCP.SRC]NCPNET10.B32;1
Page 10
(6)

00000000V	00	62	DD	0001A	PUSHL	NCP\$GL EXELCB	:	0325
		01	FB	0001C	CALLS	#1, NCP\$BLDLCB	:	
00000000V	00	62	DD	00023	PUSHL	NCP\$GL EXELCB	:	0326
	50	01	FB	00025	CALLS	#1, NCP\$OPENLINK	:	
		01	DD	0002C	MOVL	#1, R0	:	0327
		04	DD	0002F	RET		:	0329

; Routine Size: 48 bytes, Routine Base: \$CODE\$ + 0045

```

336 0330 1 %SBTTL 'ACT$VRB_CLEEXEC Close Link to the Executor'
337 0331 1 GLOBAL ROUTINE ACT$VRB_CLEEXEC =
338 0332 1
339 0333 1 ++
340 0334 1 FUNCTIONAL DESCRIPTION:
341 0335 1
342 0336 1 This is an action routine which closes the link to the current
343 0337 1 executor and opens a link to NML on the local node.
344 0338 1 The local node is known as '::' so we use the obj spec only
345 0339 1 to open a link to NML.
346 0340 1
347 0341 1 FORMAL PARAMETERS:
348 0342 1
349 0343 1 NONE
350 0344 1
351 0345 1 IMPLICIT INPUTS:
352 0346 1
353 0347 1 NCP$GT_EXECLCB LCB to be used for the executor
354 0348 1 NCP$GL_OLDLCB Pointer to lcb for tell exec
355 0349 1 NCP$GL_EXELCB Pointer to lcb for exec
356 0350 1
357 0351 1 IMPLICIT OUTPUTS:
358 0352 1
359 0353 1 NONE
360 0354 1
361 0355 1 ROUTINE VALUE:
362 0356 1 COMPLETION CODES:
363 0357 1
364 0358 1 Success or error signaled
365 0359 1
366 0360 1 SIDE EFFECTS:
367 0361 1
368 0362 1 NONE
369 0363 1
370 0364 1 --
371 0365 1
372 0366 2 BEGIN
373 0367 2
374 0368 2 LOCAL
375 0369 2 LCB : REF BBLOCK [LCB$C_SIZE], ! Address of the lcb to be used
376 0370 2 PTR ! General pointer
377 0371 2 ;
378 0372 2
379 0373 2 NCP$GL_OLDLCB = 0; ! No tell is active
380 0374 2
381 0375 2 LCB = NCP$GT_EXECLCB; ! The lcb of interest
382 0376 2 NCP$GL_EXELCB = .LCB; ! The widely used pointer to it
383 0377 2 NCP$CLOSELINK (.LCB); ! Close it if its active
384 0378 2
385 0379 2
386 0380 2
387 0381 2 Set a pointer to the NCB and put the obj spec on. The local node
388 0382 2 will be used since we are using no node name and :: only is always
389 0383 2 the local node.
390 0384 2 Note we are using no access control, so the default access will be
391 0385 2 used for the object
392 0386 2
```

```

: 393      0387 2
: 394      0388 2      LCB [LCB$L_NCBPTR] = LCB [LCB$T_NCB];
: 395      0389 2
: 396      0390 2      PTR = .LCB [LCB$L_NCBPTR];
: 397      0391 2
: 398      0392 2      PTR = CH$MOVE
: 399      0393 2      (
: 400      0394 2          .BBLOCK [NCP$Q_OBJSPEC, DSC$W_LENGTH],
: 401      0395 2          .BBLOCK [NCP$Q_OBJSPEC, DSC$A_POINTER],
: 402      0396 2          .PTR
: 403      0397 2      );
: 404      0398 2
: 405      0399 2      LCB [LCB$L_NCBCNT] = .PTR - LCB [LCB$T_NCB];
: 406      0400 2      LCB [LCB$B_STS] = 0;
: 407      0401 2
: 408      0402 2
: 409      C'03 2      ! Link will be opened on first write
: 410      0404 2      RETURN SUCCESS
: 411      0405 1      END;
```

			01FC 00000	.ENTRY	ACT\$VRB_CLEEXEC, Save R2,R3,R4,R5,R6,R7,R8	: 0331
	58	00000000'	00 9E 00002	MOVAB	NCP\$GL_OLDLCB, R8	: 0373
			68 D4 00009	CLRL	NCP\$GL_OLDLCB	: 0375
	56	FF10	C8 9E 0000B	MOVAB	NCP\$GT_EXECLCB, LCB	: 0376
04	A8		56 D0 00010	MOVL	LCB, NCP\$GL_EXELCB	: 0377
			56 DD 00014	PUSHL	LCB	: 0388
00000000V	00		01 FB 00016	CALLS	#1, NCP\$CLOSELINK	: 0390
	57	12	A6 9E 0001D	MOVAB	18(R6), R7	: 0395
OE	A6		57 D0 00021	MOVL	R7, 14(LCB)	: 0396
	53	OE	A6 D0 00025	MOVL	14(LCB), PTR	: 0399
	50	00000000'	00 D0 00029	MOVL	NCP\$Q_OBJSPEC+4, R0	: 0400
	60	00000000'	00 28 00030	MOVC3	NCP\$Q_OBJSPEC, (R0), (PTR)	: 0403
OA	63		57 C3 00038	SUBL3	R7, PTR, 10(LCB)	: 0405
A6			66 94 0003D	CLRB	(LCB)	
	50		01 D0 0003F	MOVL	#1, R0	
			04 00042	RET		

; Routine Size: 67 bytes, Routine Base: \$CODE\$ + 0075


```

413 0406 1 %SBTTL 'NCP$BLDLCB Build an Link Control Block'
414 0407 1 ROUTINE NCP$BLDLCB (LCB) :NOVALUE = !
415 0408 1
416 0409 1
417 0410 1 ++
418 0411 1 FUNCTIONAL DESCRIPTION:
419 0412 1 This routine builds the contents of an LCB (link control block)
420 0413 1 from information in left around by the parse.
421 0414 1 The nodename which may be a logical name, is translated 10 times or
422 0415 1 until it does not translate further, which ever is first.
423 0416 1 If access control is provided with the node spec, it is appended to
424 0417 1 the translation after any access control is stripped from the
425 0418 1 translation. If no access control is provided in the node spec,
426 0419 1 it may be specified in the logical. The logical name cannot contain
427 0420 1 ::. The translation may or may not contain ::.
428 0421 1
429 0422 1 FORMAL PARAMETERS:
430 0423 1
431 0424 1 LCB Address of the link control block
432 0425 1
433 0426 1 IMPLICIT INPUTS:
434 0427 1
435 0428 1 PDB$G_VRB_XID Node spec string
436 0429 1 ACT$GQ_ACCACC_DSC Descriptors of access control
437 0430 1 ACT$GQ_ACCPSW_DSC
438 0431 1 ACT$GQ_ACCUSR_DSC
439 0432 1 ACT$GL_XIDACC_Q True for access control in node spec
440 0433 1
441 0434 1 IMPLICIT OUTPUTS:
442 0435 1
443 0436 1 NONE
444 0437 1
445 0438 1 ROUTINE VALUE:
446 0439 1 COMPLETION CODES:
447 0440 1
448 0441 1 NONE
449 0442 1
450 0443 1 SIDE EFFECTS:
451 0444 1
452 0445 1 NONE
453 0446 1
454 0447 1 --
455 0448 1
456 0449 2 BEGIN
457 0450 2
458 0451 2 MAP
459 0452 2 LCB : REF BBLOCK ! Pointer to an link control block
460 0453 2 ;
461 0454 2
462 0455 2 LITERAL
463 0456 2 RSLSZ = 64 ! Size for tranlation buffer
464 0457 2 ;
465 0458 2
466 0459 2 LOCAL
467 0460 2 RSLBUF : VECTOR [RSLSZ, BYTE], ! Translation buffer
468 0461 2 RSLDSC : VECTOR [2], ! Descriptor of buffer
469 0462 2 RSDSC : VECTOR [2], ! Descriptor of whole buffer
```

```

470 0463 2 STATUS,
471 0464 2 ACCPTR,
472 0465 2 ACCCNT,
473 0466 2 PTR,
474 0467 2 CTR,
475 0468 2 ;
476 0469 2
477 0470 2 EXTERNAL LITERAL
478 0471 2 NCP$_INVACC
479 0472 2 ;
480 0473 2
481 0474 2 EXTERNAL
482 0475 2 ACT$GQ_ACCACC_DSC,
483 0476 2 ACT$GQ_ACCPSW_DSC,
484 0477 2 ACT$GQ_ACCUSR_DSC,
485 0478 2 ACT$GL_XIDACC_Q,
486 0479 2 PDB$G_VRB_XID
487 0480 2 ;
488 0481 2
489 0482 2
490 0483 2
491 0484 2 Obtain the node spec and strip trailing colons
492 0485 2
493 0486 2
494 0487 2 PTR = BBLOCK [PDB$G_VRB_XID, PDB$_DATA];
495 0488 2 CTR = CH$RCHAR_A (PTR);
496 0489 2
497 0490 2 DEGRA IDX FROM .PTR + .CTR - 1
498 0491 2 TO .PTR
499 0492 2 DO
500 0493 2 IF CH$RCHAR (.IDX) EQL ':'
501 0494 2 THEN CTR = .CTR - 1
502 0495 2 ELSE EXITLOOP
503 0496 2 ;
504 0497 2
505 0498 2 CH$MOVE (.CTR, .PTR, RSLBUF);
506 0499 2 RSLDSC [0] = .CTR;
507 0500 2 RSLDSC [1] = RSLBUF;
508 0501 2 RSLDSC [1] = RSLBUF;
509 0502 2 RSLDSC [0] = RSLSZ;
510 0503 2
511 0504 2 IF .ACT$GL_XIDACC_Q
512 0505 2 THEN
513 0506 2 BEGIN
514 0507 2 ACCPTR = CH$FIND_CH (.CTR, .PTR, '');
515 0508 2 RSLDSC [0] = .ACCPTR - .PTR;
516 0509 2 ACCCNT = .CTR - .RSLDSC [0]
517 0510 2 END
518 0511 2 ;
519 0512 2
520 0513 2 DEGRU IDX FROM 10 TO 1
521 0514 2 DO
522 0515 2 BEGIN
523 0516 2 STATUS = $TRNLOG
524 0517 2 (
525 0518 2 LOGNAM = RSLDSC,
526 0519 2 RSLLEN = RSLDSC [0],

```

```

! Return status of translation
! Pointer to original access control
! Size of access control

```

```

! Invalid access control signal

```

```

! Descriptors for access control

```

```

! Access control present in nodespec
! Nodespec counted string here

```

```

! Obtain node spec string
! And its size

```

```

! Strip off trailing :: to
! Begin translation

```

```

! Copy to result buffer
! Build descriptor
! Describe whole buffer too

```

```

! If Access control specified
! Strip it off before trans

```

```

! Find it
! Shorten descriptor
! Size of our access control

```

```

! Translate logical 10 deep

```

```

! Obtain one translation

```

```

! Here is the name to trans
! Return the length here

```



```

: 527      P 0520 3          RSLBUF = RSDSC          ! Return the string here
: 528      0521 4          )
: 529      0522 4
: 530      0523 4          IF NOT .STATUS          ! If any error
: 531      0524 4          OR
: 532      0525 4          .STATUS EQL SSS_NOTRAN    ! or no translation
: 533      0526 4          THEN EXITLOOP          ! we are done
: 534      0527 4          END
: 535      0528 4          ;
: 536      0529 4
: 537      0530 4          IF .ACT$GL_XIDACC_Q      ! If node spec had acc control
: 538      0531 4          THEN                    ! Use as override
: 539      0532 4          BEGIN
: 540      0533 4          PTR = CH$FIND CH (.RSLDSC [0], .RSLDSC [1], '');
: 541      0534 4          IF CH$FAIL (.PTR)        ! If no acc in logical
: 542      0535 4          THEN
: 543      0536 4          PTR = .RSLDSC [1] + .RSLDSC [0] ! Add ours at end
: 544      0537 4          ;
: 545      0538 4          PTR = CH$MOVE (.ACCCNT, .ACCPTR, .PTR); ! Add our acc ctl at end of
: 546      0539 4          RSLDSC [0] = .PTR - .RSLDSC [1] ! translation
: 547      0540 4          END
: 548      0541 4          ;
: 549      0542 4
: 550      0543 4          PTR = LCB [LCBST_NCB];    ! Set pointer to start
: 551      0544 4          CH$MOVE                  ! Copy node spec to lcb
: 552      0545 4          (
: 553      0546 4          .RSLDSC [0],
: 554      0547 4          .RSLDSC [1],
: 555      0548 4          .PTR
: 556      0549 4          );
: 557      0550 4          CTR = .RSLDSC [0];        ! Set the counter for it
: 558      0551 4
: 559      0552 4          DECRA IDX FROM .PTR + .CTR - 1 ! Strip the colons again
: 560      0553 4          TO .PTR                  ! just to be sure
: 561      0554 4          DO
: 562      0555 4          IF CH$RCHAR (.IDX) EQL ':'
: 563      0556 4          THEN CTR = .CTR - 1
: 564      0557 4          ELSE EXITLOOP
: 565      0558 4          ;
: 566      0559 4
: 567      0560 4          Obtain the access control if its needed
: 568      0561 4
: 569      0562 4
: 570      0563 4
: 571      0564 4          PTR = LCB [LCBST_NCB] + .CTR; ! Point to the copied string
: 572      0565 4
: 573      0566 4          IF .ACT$GL_XIDACC_Q      ! Is there access control in
: 574      0567 4          THEN                    ! The node spec?
: 575      0568 4          BEGIN
: 576      0569 4          IF .ACT$GQ_ACCACC_DSC NEQ 0 ! If so, there must not be
: 577      0570 4          OR                        ! Access control elsewhere
: 578      0571 4          .ACT$GQ_ACCPSW_DSC NEQ 0
: 579      0572 4          OR
: 580      0573 4          .ACT$GQ_ACCUSR_DSC NEQ 0
: 581      0574 4          THEN
: 582      0575 4          SIGNAL_STOP (NCP$_INVACC) ! Signal too much access ctl
: 583      0576 4          END
```

```
584 0577 2 ELSE
585 0578 3 BEGIN
586 0579 3 IF .ACT$GQ_ACCUSR_DSC NEQ 0 ! If not, use other access ctl
587 0580 3 THEN
588 0581 4 BEGIN ! Look through the name we
589 0582 4 ACCPTR = CH$FIND_CH (.CTR, LCB [LCB$T_NCB], ''); ! for acc ctl
590 0583 4 IF NOT CH$FAIL (.ACCPTR)
591 0584 4 THEN
592 0585 4 PTR = .ACCPTR
593 0586 4 ;
594 0587 4 CH$WCHAR_A ('', PTR); ! Put it in standard form
595 0588 4 PTR = CH$MOVE
596 0589 4 (
597 0590 4 .BBLOCK [ACT$GQ_ACCUSR_DSC, DSC$W_LENGTH],
598 0591 4 .BBLOCK [ACT$GQ_ACCUSR_DSC, DSC$A_POINTER],
599 0592 4 .PTR
600 0593 4 );
601 0594 4
602 0595 4 IF .ACT$GQ_ACCPSW_DSC NEQ 0 ! A password??
603 0596 4 THEN
604 0597 5 BEGIN
605 0598 5 CH$WCHAR_A (' ', PTR);
606 0599 5 PTR = CH$MOVE
607 0600 5 (
608 0601 5 .BBLOCK [ACT$GQ_ACCPSW_DSC, DSC$W_LENGTH],
609 0602 5 .BBLOCK [ACT$GQ_ACCPSW_DSC, DSC$A_POINTER],
610 0603 5 .PTR
611 0604 5 )
612 0605 5 END
613 0606 4 ELSE
614 0607 4 SIGNAL_STOP (NCP$_INVACC) ! If no password, not complete
615 0608 4 ;
616 0609 4
617 0610 4 IF .ACT$GQ_ACCACC_DSC NEQ 0 ! An account??
618 0611 4 THEN
619 0612 5 BEGIN
620 0613 5 CH$WCHAR_A (' ', PTR);
621 0614 5 PTR = CH$MOVE
622 0615 5 (
623 0616 5 .BBLOCK [ACT$GQ_ACCACC_DSC, DSC$W_LENGTH],
624 0617 5 .BBLOCK [ACT$GQ_ACCACC_DSC, DSC$A_POINTER],
625 0618 5 .PTR
626 0619 5 )
627 0620 5 END
628 0621 4 ;
629 0622 4
630 0623 4 CH$WCHAR_A ('', PTR); ! End the access control spec
631 0624 4
632 0625 4 END
633 0626 3 END
634 0627 2 ;
635 0628 2
636 0629 2 Copy the object connect specification to the end
637 0630 2
638 0631 2
639 0632 2
640 0633 2 PTR = CH$MOVE
```

NCPNETIO
V04-000

Network I/O Routines
NCP\$BLDLCB Build an Link Control Block

M 16
15-Sep-1984 23:46:44
14-Sep-1984 12:48:14

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[NCP.SRC]NCPNETIO.B32;1 Page 17 (8)

```

: 641      0634      2      (
: 642      0635      2          .BBLOCK [NCP$Q_OBJSPEC, DSC$W_LENGTH],
: 643      0636      2          .BBLOCK [NCP$Q_OBJSPEC, DSC$A_POINTER],
: 644      0637      2          .PTR
: 645      0638      2          );
: 646      0639      2
: 647      0640      2      !
: 648      0641      2      Fill up the LCB pointers and status
: 649      0642      2      !
: 650      0643      2
: 651      0644      2      LCB [LCB$L_NCBENT] = .PTR - LCB [LCB$T_NCB];
: 652      0645      2      LCB [LCB$L_NCBPTR] = LCB [LCB$T_NCB];
: 653      0646      2      LCB [LCB$B_STS] = 0;
: 654      0647      2
: 655      0648      2      RETURN
: 656      0649      2
: 657      0650      1      END;
```

```

.EXTRN NCP$ INVACC, ACT$GQ_ACCACC_DSC
.EXTRN ACT$GQ_ACCPSW_DSC
.EXTRN ACT$GQ_ACCUSR_DSC
.EXTRN ACT$GL_XIDACC_Q
.EXTRN PDB$G_VRB_XID, SYS$TRNLOG
```

OFFC 00000 NCP\$BLDLCB:

		5B	00000000G	00	9E	00002	.WORD	Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11	: 0407		
		5E	B0	AE	9E	00009	MOVAB	ACT\$GQ_ACCACC_DSC, R11			
		57	00000000G	00	9E	0000D	MOVAB	-80(SP), SP			
		58		87	9A	00014	MOVZBL	PDB\$G_VRB_XID+1, PTR	: 0487		
		57		58	C1	00017	ADDL3	(PTR)+, CTR	: 0488		
				07	11	0001B	BRB	CTR, PTR, R0	: 0490		
		3A		60	91	0001D	1\$: CMPB	2\$, (IDX), #58	: 0493		
				09	12	00020	BNEQ	3\$			
				58	D7	00022	DECL	CTR	: 0494		
				50	D7	00024	2\$: DECL	IDX	: 0493		
		57		50	D1	00026	CMPL	IDX, PTR			
				F2	1E	00029	BGEQU	1\$			
		67		58	28	0002B	3\$: MOVC3	CTR, (PTR), RSLBUF	: 0498		
10	AE			58	D0	00C30	MOVL	CTR, RSLDSC	: 0499		
		08	AE	10	AF	9E	00034	MOVAB	RSLBUF, RSLDSC+4	: 0500	
		0C	AE	10	AE	9E	00039	MOVAB	RSLBUF, RSLDSC+4	: 0501	
		04	AE	40	8F	9A	0003E	MOVZBL	#64, RSLDSC	: 0502	
		6E		15	00000000G	00	E9	00042	BLBC	: 0504	
				58	22	3A	00049	LOCC	ACT\$GL_XIDACC_Q, 5\$	: 0507	
		67			02	12	0004D	BNEQ	4\$		
				51	D4	0004F	CLRL	R1			
				59	51	D0	00051	4\$: MOVL	R1, ACCPTR		
		08	AE	59	57	C3	00054	SUBL3	PTR, ACCPTR, RSLDSC	: 0508	
		53		58	08	AE	C3	00059	SUBL3	RSLDSC, CTR, ACCCNT	: 0509
				52	0A	D0	0005E	5\$: MOVL	#10, IDX	: 0513	
					7E	7C	00061	6\$: CLRQ	-(SP)	: 0521	
					7E	D4	00063	CLRL	-(SP)		
					0C	AE	9F	00065	PUSHAB	RSBDSC	
				18	AE	9F	00068	PUSHAB	RSLDSC		

			1C	AE	9F	0006B	PUSHAB	RSLDSC		
		00000000G	00	06	FB	0006E	CALLS	#6, SYS\$TRNLOG		
			0D	50	E9	00075	BLBC	STATUS, 7\$	0523	
		00000629	8F	50	D1	00078	CMPL	STATUS, #1577	0525	
				04	13	0007F	BEQL	7\$		
				52	D7	00081	DECL	IDX	0523	
				DC	12	00083	BNEQ	6\$		
			5A	00	D0	00085	7\$:	MOVL	ACT\$GL_XIDACC_Q, R10	0530
			22	5A	E9	0008C		BLBC	R10, 10\$	
0C	BE	08	AE	22	3A	0008F		LOCC	#34, RSLDSC, @RSLDSC+4	0533
				02	12	00095		BNEQ	8\$	
			57	51	D4	00097		CLRL	R1	
				51	D0	00099	8\$:	MOVL	R1, PTR	
				06	12	0009C		BNEQ	9\$	0534
	57	0C	AE	08	AE	C1		ADDL3	RSLDSC, RSLDSC+4, PTR	0536
	67		69	53	28	000A4	9\$:	MOVC3	ACCCNT, (ACCPTR), (PTR)	0538
			57	53	D0	000A8		MOVL	R3, PTR	
08	AE		57	0C	AE	C3		SUBL3	RSLDSC+4, PTR, RSLDSC	0539
			56	04	AC	D0	10\$:	MOVL	LCB, R6	0543
			57	12	A6	9E		MOVAB	18(R6), PTR	
	67	0C	BE	08	AE	28		MOVC3	RSLDSC, @RSLDSC+4, (PTR)	0548
			58	08	AE	D0		MOVL	RSLDSC, CTR	0550
50			57		58	C1		ADDL3	CTR, PTR, R0	0552
				07	11	000C7		BRB	12\$	
			3A	60	91	000C9	11\$:	CMPB	(IDX), #58	0555
				09	12	000CC		BNEQ	13\$	
				58	D7	000CE		DECL	CTR	0556
				50	D7	000D0	12\$:	DECL	IDX	0555
			57	50	D1	000D2		CMPL	IDX, PTR	
				F2	1E	000D5		BGEQU	11\$	
			57	12	A846	9E	13\$:	MOVAB	18(CTR)[R6], PTR	0564
			23	5A	E9	000DC		BLBC	R10, 15\$	0566
				6B	D5	000DF		TSTL	ACT\$GQ_ACCACC_DSC	0569
				10	12	000E1		BNEQ	14\$	
				00	D5	000E3		TSTL	ACT\$GQ_ACCPSW_DSC	0571
				08	12	000E9		BNEQ	14\$	
				00	D5	000EB		TSTL	ACT\$GQ_ACCUSR_DSC	0573
				7E	13	000F1		BEQL	21\$	
				8F	DD	000F3	14\$:	PUSHL	#NCP\$ INVACC	0575
		00000000G	00	01	FB	000F9		CALLS	#1, LIB\$STOP	
				6F	11	00100		BRB	21\$	0568
				00	D5	00102	15\$:	TSTL	ACT\$GQ_ACCUSR_DSC	0579
				67	13	00108		BEQL	21\$	
12	A6		58	22	3A	0010A		LOCC	#34, CTR, 18(R6)	0582
				02	12	0010F		BNEQ	16\$	
				51	D4	00111		CLRL	R1	
			59	51	D0	00113	16\$:	MOVL	R1, ACCPTR	
				03	13	00116		BEQL	17\$	0583
			57	59	D0	00118		MOVL	ACCPTR, PTR	0585
			87	22	90	0011B	17\$:	MOVB	#34, (PTR)+	0587
			50	00	D0	0011E		MOVL	ACT\$GQ_ACCUSR_DSC+4, R0	0591
	67		60	00	28	00125		MOVC3	ACT\$GQ_ACCUSR_DSC, (R0), (PTR)	0592
			57	53	D0	0012D		MOVL	R3, PTR	
				00	D5	00130		TSTL	ACT\$GQ_ACCPSW_DSC	0595
				17	13	00136		BEQL	18\$	
			87	20	90	00138		MOVB	#32, (PTR)+	0598
			50	00	D0	0013B		MOVL	ACT\$GQ_ACCPSW_DSC+4, R0	0602

NCPNETIO
V04-000

Network I/O Routines
NCP\$BLDLCB Build an Link Control Block

C 1
15-Sep-1984 23:46:44
14-Sep-1984 12:48:14

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

Page 19
(8)

NC
VO

67	60	00000000CG	00	28	00142	MOV C3	ACT\$GQ_ACCPSW_DSC, (R0), (PTR)	: 0603
	57		53	D0	0014A	MOVL	R3, PTR	: 0599
		00000000G	00	0D	11 0014D	BRB	19\$	: 0607
			8F	DD	0014F	PUSHL	#NCP\$ INVACC	: 0610
			01	FB	00155	CALLS	#1, LIB\$STOP	: 0613
			6B	D5	0015C	TSTL	ACT\$GQ_ACCACC_DSC	: 0617
			0E	13	0015E	BEQL	20\$	: 0618
	87		20	90	00160	MOVB	#32, (PTR)+	: 0623
	50	04	AB	D0	00163	MOVL	ACT\$GQ_ACCACC_DSC+4, R0	: 0636
67	60		6B	28	00167	MOV C3	ACT\$GQ_ACCACC_DSC, (R0), (PTR)	: 0637
	57		53	D0	0016B	MOVL	R3, PTR	: 0644
	87		22	90	0016E	MOVB	#34, (PTR)+	: 0645
	50	00000000'	00	D0	00171	MOVL	NCP\$Q_OBJSPEC+4, R0	: 0646
67	60	00000000'	00	28	00178	MOV C3	NCP\$Q_OBJSPEC, (R0), (PTR)	: 0650
	57		53	D0	00180	MOVL	R3, PTR	
	50	12	A6	9E	00183	MOVAB	18(R6), R0	
0A	A6		50	C3	00187	SUBL3	R0, PTR, 10(R6)	
	57		A6	9E	0018C	MOVAB	18(R6), 14(R6)	
			66	94	00191	CLRB	(R6)	
			04	00193	RET			

; Routine Size: 404 bytes, Routine Base: \$CODE\$ + 00B8

```

659 0651 1 %SBTTL 'NCP$OPENLINK Open a link to NML'
660 0652 1 GLOBAL ROUTINE NCP$OPENLINK (LCB) :NOVALUE =
661 0653 1
662 0654 1 ++
663 0655 1 FUNCTIONAL DESCRIPTION:
664 0656 1
665 0657 1 This routine opens a link to NML given an LCB address and
666 0658 1 verifies the connect data to determine if NML is phase II or
667 0659 1 phase III. The lcb already contains the NCB built in a previous
668 0660 1 step.
669 0661 1
670 0662 1 FORMAL PARAMETERS:
671 0663 1
672 0664 1 LCB Address of the LCB to use
673 0665 1
674 0666 1 IMPLICIT INPUTS:
675 0667 1
676 0668 1 NONE
677 0669 1
678 0670 1 IMPLICIT OUTPUTS:
679 0671 1
680 0672 1 NONE
681 0673 1
682 0674 1 ROUTINE VALUE:
683 0675 1 COMPLETION CODES:
684 0676 1
685 0677 1 NONE errors signaled
686 0678 1
687 0679 1 SIDE EFFECTS:
688 0680 1
689 0681 1 NONE
690 0682 1
691 0683 1 --
692 0684 1
693 0685 2 BEGIN
694 0686 2
695 0687 2 LITERAL
696 0688 2 MBXSIZ = 10 ! Max size of mailbox name
697 0689 2 ;
698 0690 2
699 0691 2 MAP
700 0692 2 LCB : REF BBLOCK ! The link control block
701 0693 2 ;
702 0694 2
703 0695 2 LOCAL
704 0696 2 MBXBUF : VECTOR [MBXSIZ, BYTE], ! Buffer to build mailbox name
705 0697 2 MBXLST : VECTOR [2], ! FAO list for mailbox name
706 0698 2 MBXDSC : VECTOR [2], ! Descriptor of mailbox name buffer
707 0699 2 IOSB : BBLOCK [8], ! IO status block
708 0700 2 STATUS, ! Return status
709 0701 2 PTR, ! General pointer
710 0702 2 CTR ! General counter
711 0703 2 ;
712 0704 2
713 0705 2 OWN
714 0706 2 CHNCHAR : BBLOCK [DIB$K_LENGTH] ! Channel characteristics
715 0707 2 ;
```



```

716 0708 2
717 0709 2
718 0710 2      EXTERNAL LITERAL
719 0711 2      NCP$_CONNEC,      ! Connect errors
720 0712 2      NCP$_UNSVRS      ! Unsupported version of nml
721 0713 2      ;
722 0714 2      LCB [LCB$_MBXCHN] = 0;      ! Make the channels zero
723 0715 2      LCB [LCB$_CHAN] = 0;      ! to indicate they are not here
724 0716 2
725 0717 2      LCB [LCB$_STS] = TRUE;      ! This lcb is now open
726 0718 2
727 0719 2      LCB [LCB$_PH2] = FALSE;      ! Assume Phase III
728 0720 2
729 0721 2      CH$FILL(0, 3, LCB [LCB$_NMLVERS]); ! Preset NML version to null
730 0722 2
731 0723 2
732 0724 2      If we are going to communicate with the NML on the local node,
733 0725 2      and there is no access control string, then establish communications
734 0726 2      with the sharable version of NML linked with this program, rather
735 0727 2      than starting up another NML process on this node.
736 0728 2
737 0729 2
738 0730 2      IF CH$RCHAR(.LCB [LCB$_NCBPTR]) EQL ':'
739 0731 2      THEN
740 0732 2      BEGIN
741 0733 2      NML$INITIALIZE();      ! Initialize NICE processor
742 0734 2      CH$MOVE(3, UPLIT BYTE(NCP$_VRS, NCP$_ECO, NCP$_UECO),
743 0735 2      LCB [LCB$_NMLVERS]);      ! Assume NMLSHR is same as our version
744 0736 2      RETURN;      ! Return successfully
745 0737 2      END;
746 0738 2
747 0739 2      We are about to do a non-transparent connect, so first
748 0740 2      we must create a mailbox.
749 0741 2
750 0742 2
751 P 0743 2      STATUS = $CREMBX
752 P 0744 2      (
753 P 0745 2      CHAN = LCB [LCB$_MBXCHN],
754 P 0746 2      MAXMSG = 64,
755 P 0747 2      BUFQUO = 256,
756 P 0748 2      PROMSK = 'XX'FF00'      ! own-sys=rwed
757 0749 2      );
758 0750 2      NCP$SIGNETERR (NCP$_CONNEC, .STATUS, 0);      ! Signal the error
759 0751 2
760 P 0752 2      STATUS = $GETCHN      ! Obtain the mailbox name
761 P 0753 2      (
762 P 0754 2      CHAN = .LCB [LCB$_MBXCHN],
763 P 0755 2      PRIBUF = UPLIT (DISK$_LENGTH, CHNCHAR)
764 0756 2      );
765 0757 2      NCP$SIGNETERR (NCP$_CONNEC, .STATUS, 0);      ! Report an error
766 0758 2
767 0759 2      PTR = .CHNCHAR [DISK$_DEVNAMOFF];      ! Offset to the name
768 0760 2      IF .PTR EQL 0      ! Zero means missing
769 0761 2      THEN      ! No name, so we die here
770 0762 2      NCP$SIGNETERR (NCP$_CONNEC, SS$_IVCHAN, 0)
771 0763 2      ;
772 0764 2
```

```

773 0765 2 MBXLST [0] = CHNCHAR + .PTR; ! Data list has pointer to the name
774 0766 2 MBXLST [1] = .CHNCHAR [DIB$W_UNIT]; ! The unit number to convert
775 0767 2 MBXDSC [0] = MBXSIZ; ! Build descriptor of buffer, size and
776 0768 2 MBXDSC [1] = MBXBUF; ! Address of the buffer
777 0769 2
778 P 0770 2 $FAOL ! Build the whole mailbox name
779 P 0771 2 (
780 P 0772 2 CTRSTR = ASCID ('!AC!UW:'), ! The name and unit MBAnnn:
781 P 0773 2 OUTLEN = MBXDSC [0], ! Length goes back in descriptor
782 P 0774 2 OUTBUF = MBXDSC, ! Descriptor is here
783 P 0775 2 PRMLST = MBXLST ! Data list is here
784 0776 2 );
785 0777 2
786 P 0778 2 STATUS = $ASSIGN ! Assign a channel to the network
787 P 0779 2 (
788 P 0780 2 DEVNAM = ASCID ('_NET:'), ! General device for network
789 P 0781 2 CHAN = LCB [LCB$W_CHAN], ! Place to put channel number
790 P 0782 2 MBXNAM = MBXDSC ! Name we built with FAO
791 0783 2 );
792 0784 2 NCP$SIGNETERR (NCP$_CONNEC, .STATUS, 0); ! Report an error
793 0785 2
794 P 0786 2 STATUS = $QIOW ! Create a logical link to NML
795 P 0787 2 (
796 P 0788 2 CHAN = .LCB [LCB$W_CHAN], ! Use network channel
797 P 0789 2 FUNC = IOS$ ACCESS, ! ACP function
798 P 0790 2 IOSB = IOSB, ! Status here
799 P 0791 2 P2 = LCB [LCB$L_NCBCNT] ! This is the NCB descriptor
800 0792 2 );
801 0793 2 NCP$SIGNETERR (NCP$_CONNEC, .STATUS, IOSB); ! An error
802 0794 2
803 P 0795 2 STATUS = $QIOW ! Read the connect data
804 P 0796 2 (
805 P 0797 2 CHAN = .LCB [LCB$W_MBXCHN], ! Channel for mailbox
806 P 0798 2 FUNC = IOS$ READVBLR,
807 P 0799 2 IOSB = IOSB,
808 P 0800 2 P1 = NCP$GT_MBXBFR, ! Read data into mailbox buffer
809 P 0801 2 P2 = NCP$C_MBXSI2
810 0802 2 );
811 0803 2 NCP$SIGNETERR (NCP$_CONNEC, .STATUS, IOSB);
812 0804 2
813 0805 2 !
814 0806 2 Validate the message and its returned optional data
815 0807 2 !
816 0808 2
817 0809 2 STATUS = .BBLOCK [NCP$GT_MBXBFR, 0,0,16,0];
818 0810 2 PTR = NCP$GT_MBXBFR + 4;
819 0811 2
820 0812 2 IF .STATUS NEQ MSG$ CONFIRM ! It must be a connect confirm
821 0813 2 THEN SIGNAL_STOP (NCP$_CONNEC) ! Otherwise blow away
822 0814 2 ;
823 0815 2
824 0816 2 CTR = .IOSB [2, 0, 16, 0] - 4; ! Play games to look at the data
825 0817 2 CTR = .CTR - CH$RCHAR (.PTR) - 1; ! Skip over the device name
826 0818 2 PTR = .PTR + CH$RCHAR (.PTR) + 1;
827 0819 2
828 0820 2 IF CH$RCHAR (.PTR) LEQ 0 ! Any data returned?
829 0821 2 THEN LCB [LCB$B_PH2] = TRUE ! No, its phase II
```



```
.PSECT SPLITS,NOWRT,NOEXE,2
```

```

00 00 04 00024 P.AAC: .BYTE 4, 0, 0
00027 .BLKB 1
00000074 00028 P.AAD: .LONG 116
00000000' 0002C .ADDRESS CHNCHAR
3A 57 55 21 43 41 21 5F 00030 P.AAF: .ASCII \-!AC!UW:\
00000008 00038 P.AAE: .LONG 8
00000000' 0003C .ADDRESS P.AAF
00 00 00 3A 54 45 4E 5F 00040 P.AAH: .ASCII \-NET:\<0><0><0>
00000005 00048 P.AAG: .LONG 5
00000000' 0004C .ADDRESS P.AAH
00 00 04 00050 P.AAI: .BYTE 4, 0, 0

```

```
00 00 02 00053 .BLKB 1
00 00 02 00054 P.AAJ: .BYTE 2, 0, 0
00 00 03 00057 .BLKB 1
00 00 03 00058 P.AAK: .BYTE 3, 0, 0
```

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

```
00008 CHNCHAR: .BLKB 116
```

```
.EXTRN NCP$ CONNEX, NCP$ UNSVRS
.EXTRN SYSS$CREMBX, SYSS$GETCHN
.EXTRN SYSS$FAOL, SYSS$ASSIGN
.EXTRN SYSS$QIOW
```

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

```
OFFC 00000
```

```
.ENTRY NCP$OPENLINK, Save R2,R3,R4,R5,R6,R7,R8,R9,-; 0652
R10,R11
```

		5B	00000000G	00	9E	00002	MOVAB	SYSS\$QIOW, R11		
		5A	00000000'	00	9E	00009	MOVAB	NCP\$GT MBXBFR, R10		
		59	00000000'	00	9E	00010	MOVAB	CHNCHAR+14, R9		
		58	00000000V	00	9E	00017	MOVAB	NCP\$SIGNETERR, R8		
		57	00000000G	8F	D0	0001E	MOVL	#NCP\$ CONNEX, R7		
		56	00000000'	00	9E	00025	MOVAB	P.AAC, R6		
		5E		24	C2	0002C	SUBL2	#36, SP		
		55		04	AC	D0 0002F	MOVL	LCB, R5		0714
				02	A5	D4 00033	CLRL	2(R5)		0715
06	A5				01	B0 00036	MOVW	#1, (R5)		0717
		65			00	F0 00039	INSV	#0, #0, #24, 6(R5)		0721
		0C			B5	91 0003F	CMPB	@14(R5), #58		0730
		3A	0E		0E	12 00043	BNEQ	1\$		
			00000000G	00	00	FB 00045	CALLS	#0, NML\$INITIALIZE		0733
06	A5			18	00	66 F0 0004C	INSV	P.AAC, #0, #24, 6(R5)		0735
						04 00052	RET			0732
					7E	7C 00053	CLRQ	-(SP)		0749
		7E	FF00		8F	3C 00055	MOVZWL	#65280, -(SP)		
		7E	0100		8F	3C 0005A	MOVZWL	#256, -(SP)		
		7E	40		8F	9A 0005F	MOVZBL	#64, -(SP)		
			04		A5	9F 00063	PUSHAB	4(R5)		
					7E	D4 00066	CLRL	-(SP)		
		00000000G	00		07	FB 00068	CALLS	#7, SYSS\$CREMBX		
			52		50	D0 0006F	MOVL	R0, STATUS		0750
					7E	D4 00072	CLRL	-(SP)		
					52	DD 00074	PUSHL	STATUS		
					57	DD 00076	PUSHL	R7		
		68			03	FB 00078	CALLS	#3, NCP\$SIGNETERR		0756
					7E	7C 0007B	CLRQ	-(SP)		
				04	A6	9F 0007D	PUSHAB	P.AAD		
					7E	D4 00080	CLRL	-(SP)		
		00000000G	7E	04	A5	3C 00082	MOVZWL	4(R5), -(SP)		
			00		05	FB 00086	CALLS	#5, SYSS\$GETCHN		
			52		50	D0 0008D	MOVL	R0, STATUS		0757
					7E	D4 00090	CLRL	-(SP)		
					52	DD 00092	PUSHL	STATUS		
					57	DD 00094	PUSHL	R7		
		68			03	FB 00096	CALLS	#3, NCP\$SIGNETERR		
		54			69	3C 00099	MOVZWL	CHNCHAR+14, PTR		0759

		0C	12	0009C	BNEQ	2\$	0760
		7E	D4	0009E	CLRL	-(SP)	0762
	7E	013C	8F	3C 000A0	MOVZWL	#316, -(SP)	
			57	DD 000A5	PUSHL	R7	
	68		03	FB 000A7	CALLS	#3, NCP\$SIGNETERR	
10	AE	F2	A9	9E 000AA	MOVAB	CHNCHAR[PTR], MBXLST	0765
14	AE	FE	A9	3C 000B0	MOVZWL	CHNCHAR+12, MBXLST+4	0766
08	AE		0A	DD 000B5	MOVL	#10, MBXDSC	0767
0C	AE		18	AE 9E 000B9	MOVAB	MBXBUF, MBXDSC+4	0768
			10	AE 9F 000BE	PUSHAB	MBXLST	0776
			0C	AE 9F 000C1	PUSHAB	MBXDSC	
			10	AE 9F 000C4	PUSHAB	MBXDSC	
			14	A6 9F 000C7	PUSHAB	P.AAE	
00000000G	00		04	FB 000CA	CALLS	#4, SY\$FAOL	
			08	AE 9F 000D1	PUSHAB	MBXDSC	0783
			7E	D4 000D4	CLRL	-(SP)	
			02	A5 9F 000D6	PUSHAB	2(R5)	
			24	A6 9F 000D9	PUSHAB	P.AAG	
00000000G	00		04	FB 000DC	CALLS	#4, SY\$ASSIGN	
52			50	DD 000E3	MOVL	R0, STATUS	
			7E	D4 000E6	CLRL	-(SP)	0784
			52	DD 000E8	PUSHL	STATUS	
			57	DD 000EA	PUSHL	R7	
	68		03	FB 000EC	CALLS	#3, NCP\$SIGNETERR	
			7E	7C 000EF	CLRL	-(SP)	0792
			7E	7C 000F1	CLRL	-(SP)	
		0A	A5	9F 000F3	PUSHAB	10(R5)	
			7E	7C 000F6	CLRL	-(SP)	
			7E	D4 000F8	CLRL	-(SP)	
		20	AE	9F 000FA	PUSHAB	IOSB	
			32	DD 000FD	PUSHL	#50	
	7E	02	A5	3C 000FF	MOVZWL	2(R5), -(SP)	
			7E	D4 00103	CLRL	-(SP)	
	68		0C	FB 00105	CALLS	#12, SY\$QIOW	
52			50	DD 00108	MOVL	R0, STATUS	
		4004	8F	BB 0010B	PUSHR	#*M<R2,SP>	0793
			57	DD 0010F	PUSHL	R7	
	68		03	FB 00111	CALLS	#3, NCP\$SIGNETERR	
			7E	7C 00114	CLRL	-(SP)	0802
			7E	7C 00116	CLRL	-(SP)	
			28	DD 00118	PUSHL	#40	
			5A	DD 0011A	PUSHL	R10	
			7E	7C 0011C	CLRL	-(SP)	
		20	AE	9F 0011E	PUSHAB	IOSB	
			31	DD 00121	PUSHL	#49	
	7E	04	A5	3C 00123	MOVZWL	4(R5), -(SP)	
			7E	D4 00127	CLRL	-(SP)	
	68		0C	FB 00129	CALLS	#12, SY\$QIOW	
52			50	DD 0012C	MOVL	R0, STATUS	
		4004	8F	BB 0012F	PUSHR	#*M<R2,SP>	0803
			57	DD 00133	PUSHL	R7	
	68		03	FB 00135	CALLS	#3, NCP\$SIGNETERR	
52			6A	3C 00138	MOVZWL	NCP\$GT_MBXBFR, STATUS	0809
54		04	AA	9E 0013B	MOVAB	NCP\$GT_MBXBFR+4, PTR	0810
31			52	D1 0013F	CMPL	STATUS, #49	0812
			09	13 00142	BEQL	3\$	
			57	DD 00144	PUSHL	R7	0813

NCPNETIO
V04-000

Network I/O Routines
NCP\$OPENLINK Open a link to NML

J 1
15-Sep-1984 23:46:44
14-Sep-1984 12:48:14

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[NCP.SRC]NCPNETIO.B32;1 Page 26
(9)

			00000000G	00		01	FB	00146		CALLS	#1, LIB\$STOP		
				50		02	AE	3C	0014D	3\$:	MOVZWL	10SB+2, CTR	0816
				50			04	C2	00151		SUBL2	#4, CTR	
				51			64	9A	00154		MOVZBL	(PTR), R1	0817
	52			50			51	C3	00157		SUBL3	R1, CTR, R2	
				50		FF	A2	9E	0015B		MOVAB	-1(R2), CTR	
				54		01	A144	9E	0015F		MOVAB	1(R1)[PTR], PTR	0818
							64	95	00164		TSTB	(PTR)	0820
							05	12	00166		BNEQ	4\$	
			01	A5			01	90	00168		MOVB	#1, 1(R5)	0821
								04	0016C		RET		
				03			64	91	0016D	4\$:	CMPB	(PTR), #3	0825
							23	12	00170		BNEQ	6\$	
	2C	A6		01	A4		03	29	00172		CMPC3	#3, 1(PTR), P.AAI	0828
							10	1E	00178		BGEQU	5\$	
	30	A6		01	A4		03	29	0017A		CMPC3	#3, 1(PTR), P.AAJ	0835
							08	13	00180		BEQL	5\$	
	34	A6		01	A4		03	29	00182		CMPC3	#3, 1(PTR), P.AAK	0842
							0B	12	00188		BNEQ	6\$	
06	A5		18		00	01	A4	F0	0018A	5\$:	INSV	1(PTR), #0, #24, 6(R5)	0849
						01	A5	94	00191		CLRB	1(R5)	0850
								04	00194		RET		0823
							55	DD	00195	6\$:	PUSHL	R5	0854
			00000000V	00			01	FB	00197		CALLS	#1, NCP\$CLOSELINK	
					00000000G		8F	DD	0019E		PUSHL	#NCP\$ UNSVRS	0855
			00000000G	00			01	FB	001A4		CALLS	#1, LIB\$STOP	
							04	001AB			RET		0862

; Routine Size: 428 bytes, Routine Base: \$CODE\$ + 024C

```

872 0863 1 %SBTTL 'NCP$SIGNETERR Signal a Network Communication Error'
873 0864 1 GLOBAL ROUTINE NCP$SIGNETERR (CODE, STATUS, IOSB) :NOVALUE = !
874 0865 1
875 0866 1 ++
876 0867 1 FUNCTIONAL DESCRIPTION:
877 0868 1
878 0869 1 This routine checks the results from a system service or QIO
879 0870 1 and signals an error with a subcode. Both the service status
880 0871 1 and the status block status is checked.
881 0872 1
882 0873 1 If there is an error on the link, it is closed if it is open.
883 0874 1 This will cause the link to be reopened automatically if another
884 0875 1 command is done.
885 0876 1
886 0877 1 FORMAL PARAMETERS:
887 0878 1
888 0879 1 CODE Value of the NCP code to signal
889 0880 1 STATUS Value of the system service status
890 0881 1 IOSB Address of the IOSB to check for error status
891 0882 1
892 0883 1 IMPLICIT INPUTS:
893 0884 1
894 0885 1 NONE
895 0886 1
896 0887 1 IMPLICIT OUTPUTS:
897 0888 1
898 0889 1 NONE
899 0890 1
900 0891 1 ROUTINE VALUE:
901 0892 1 COMPLETION CODES:
902 0893 1
903 0894 1 NONE error signaled with additional status
904 0895 1
905 0896 1 SIDE EFFECTS:
906 0897 1
907 0898 1 NONE
908 0899 1
909 0900 1 --
910 0901 1
911 0902 2 BEGIN
912 0903 2
913 0904 2 MAP
914 0905 2 IOSB : REF BBLOCK
915 0906 2 ;
916 0907 2
917 0908 2 LOCAL
918 0909 2 REPORT
919 0910 2 ;
920 0911 2
921 0912 3 IF NOT (REPORT = .STATUS) ! Look at the primary status
922 0913 2 OR
923 0914 2 NOT
924 0915 2 (
925 0916 3 IF .IOSB EQL 0 ! If there is no iosb
926 0917 3 THEN TRUE ! Always succeed
927 0918 4 ELSE (REPORT = .IOSB [0, 0, 16, 0] ) ! Or report the iosb error
928 0919 3 )
```

NCPNET10
V04-000

Network I/O Routines
NCP\$SIGNETERR Signal a Network Communication E

15-Sep-1984 23:46:44
14-Sep-1984 12:48:14

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[NCP.SRC]NCPNET10.B32;1 (10)

Page 28

```
: 929      0920 2      THEN
: 930      0921 3      BEGIN
: 931      0922 3      NCP$CLOSELINK (.NCP$GL_EXELCB);      ! Close link to mark to reopen
: 932      0923 3      SIGNAL_STOP (.CODE, 0, -.REPORT)      ! Signal the error
: 933      0924 3      END
: 934      0925 3
: 935      0926 1      END;
```

				0004 0C J00	.ENTRY	NCP\$SIGNETERR, Save R2		0864
	52	08	AC	D0 00002	MOVL	STATUS, REPORT		0912
	0C		52	E9 00006	BLBC	REPORT, 1\$		
		0C	AC	D5 00009	TSTL	IOSB		0916
			22	13 0000C	BEQL	2\$		
	52	0C	BC	3C 0000E	MOVZWL	@IOSB, REPORT		0918
	1B		52	E8 00012	BLBS	REPORT, 2\$		
			00	DD 00015	PUSHL	NCP\$GL_EXELCB		0922
00000000V	00		01	FB 0001B	CALLS	#1, NCP\$CLOSELINK		
			52	DD 00022	PUSHL	REPORT		0923
			7E	D4 00024	CLRL	-(SP)		
		04	AC	DD 00026	PUSHL	CODE		
00000000G	00		03	FB 00029	CALLS	#3, LIB\$STOP		
			04	00030	RET			0926

; Routine Size: 49 bytes, Routine Base: \$CODE\$ + 03F8


```

937 0927 1 %SBTTL 'NCP$CLOSELINK Close a Link Open in an LCB'
938 0928 1 GLOBAL ROUTINE NCP$CLOSELINK (LCB) :NOVALUE = !
939 0929 1
940 0930 1 ++
941 0931 1 FUNCTIONAL DESCRIPTION:
942 0932 1
943 0933 1 This routine closes a logical link open in an LCB.
944 0934 1 The LCB$B_STS byte is true for the link is open.
945 0935 1
946 0936 1 FORMAL PARAMETERS:
947 0937 1
948 0938 1 LCB Address of the lcb describing the link
949 0939 1
950 0940 1 IMPLICIT INPUTS:
951 0941 1
952 0942 1 NONE
953 0943 1
954 0944 1 IMPLICIT OUTPUTS:
955 0945 1
956 0946 1 NONE
957 0947 1
958 0948 1 ROUTINE VALUE:
959 0949 1 COMPLETION CODES:
960 0950 1
961 0951 1 NONE return always occurs, error signaled non-fatal
962 0952 1
963 0953 1 SIDE EFFECTS:
964 0954 1
965 0955 1 NONE
966 0956 1
967 0957 1 --
968 0958 1
969 0959 2 BEGIN
970 0960 2
971 0961 2 MAP
972 0962 2 LCB : REF BBLOCK ! Link control block
973 0963 2 ;
974 0964 2
975 0965 2 LOCAL
976 0966 2 STATUS ! Service status
977 0967 2 ;
978 0968 2
979 0969 2 EXTERNAL LITERAL
980 0970 2 NCP$_DISCON ! Disconnect error status
981 0971 2 ;
982 0972 2
983 0973 2 IF NOT .LCB [LCB$B_STS] ! If link not open, return
984 0974 2 THEN RETURN
985 0975 2 ;
986 0976 2
987 0977 2 LCB [LCB$B_STS] = FALSE; ! Mark its not open
988 0978 2
989 0979 2 IF CH$RCHAR(.LCB [LCB$L_NCBPTR]) EQL ':' ! If talking to sharable NML,
990 0980 2 THEN
991 0981 2 BEGIN
992 0982 2 BUILTIN REMQUE;
993 0983 2 LOCAL
```

```

: 994      0984      3      length,
: 995      0985      3      entry: REF VECTOR;
: 996      0986      3      NML$TERMINATE(); ! Perform sharable NML cleanups
: 997      0987      3      WHILE NOT REMQUE(.nml_resp_queue [0], entry) ! For each response in queue,
: 998      0988      3      DO
: 999      0989      4      BEGIN
1000      0990      4      length = .entry [2] + 12; ! Length of entry
1001      0991      4      LIB$FREE_VM(length, entry); ! Deallocate the entry
1002      0992      3      END;
1003      0993      3      RETURN;
1004      0994      3      END;
1005      0995      3
1006      0996      2      IF .LCB [LCB$W_CHAN] NEQ 0
1007      0997      2      THEN
1008      0998      3      BEGIN
P 1009      0999      3      STATUS = $DASSGN ! Deassign the channel to net
1010      1000      3      (CHAN = .LCB [LCB$W_CHAN]);
1011      1001      3      IF NOT .STATUS ! and report an error if so
1012      1002      3      THEN SIGNAL (NCP$DISCON, 0, .STATUS)
1013      1003      3      END
1014      1004      2      ;
1015      1005      2
1016      1006      2      IF .LCB [LCB$W_MBXCHN] NEQ 0
1017      1007      2      THEN
1018      1008      3      BEGIN
P 1009      1009      3      STATUS = $DASSGN ! Deassign mailbox channel, deleting it
1010      1010      3      (CHAN = .LCB [LCB$W_MBXCHN]);
1011      1011      3      IF NOT .STATUS ! and report the error
1012      1012      3      THEN SIGNAL (NCP$DISCON, 0, .STATUS)
1013      1013      3      END
1014      1014      2      ;
1015      1015      2
1016      1016      2      RETURN
1017      1017      2
1018      1018      1      END;
: 1028
```

.EXTRN NCP\$DISCON, SYS\$DASSGN

				007C 00000	.ENTRY NCP\$CLOSELINK, Save R2,R3,R4,R5,R6	: 0928
56	00000000G	00	9E	00002	MOVAB LIB\$SIGNAL, R6	
55	00000000G	8F	D0	00009	MOVL #NCP\$DISCON, R5	
54	00000000G	00	9E	00010	MOVAB SYS\$DASSGN, R4	
5E		08	C2	00017	SUBL2 #8, SP	
50	04	AC	D0	0001A	MOVL LCB, R0	: 0973
6D		60	E9	0001E	BLBC (R0), 4\$	
		60	94	00021	CLRB (R0)	: 0977
3A	0E	80	91	00023	CMPB @14(R0), #58	: 0979
		2B	12	00027	BNEQ 2\$	
00000000G	00	00	FB	00029	CALLS #0, NML\$TERMINATE	: 0986
50	00000000'	00	9E	00030	MOVAB NML RESP QUEUE, R0	: 0987
6E	00	80	0F	00037	REMQUE @0(R0), ENTRY	
		51	1D	0003B	BVS 4\$	
50		6E	D0	0003D	MOVL ENTRY, R0	: 0990
04	AE	08	A0	0C	ADDL3 #12, 8(R0), LENGTH	
				5E	PUSHL SP	: 0991
				DD		

00000000G	00	08	AE 9F 00048	PUSHAB	LENGTH	:	
			02 FB 0004B	CALLS	#2, LIB\$FREE_VM	:	
	52	04	DC 11 00052	BRB	1\$	:	0987
		02	AC D0 00054	MOVL	LCB, R2	:	0996
			B5 00058	TSTW	2(R2)	:	
	7E	02	16 13 0005B	BEQL	3\$	:	
	64		A2 3C 0005D	MOVZWL	2(R2), -(SP)	:	1000
	53		01 FB 00061	CALLS	#1, SYSSDASSGN	:	
	09		50 D0 00064	MOVL	R0, STATUS	:	
			53 E8 00067	BLBS	STATUS, 3\$	:	1001
			53 DD 0006A	PUSHL	STATUS	:	1002
			7E D4 0006C	CLRL	-(SP)	:	
	66		55 DD 0006E	PUSHL	R5	:	
		04	03 FB 00070	CALLS	#3, LIB\$SIGNAL	:	
			A2 B5 00073	TSTW	4(R2)	:	1006
			16 13 00076	BEQL	4\$	:	
	7E	04	A2 3C 00078	MOVZWL	4(R2), -(SP)	:	1010
	64		01 FB 0007C	CALLS	#1, SYSSDASSGN	:	
	53		50 D0 0007F	MOVL	R0, STATUS	:	
	09		53 E8 00082	BLBS	STATUS, 4\$	:	1011
			53 DD 00085	PUSHL	STATUS	:	1012
			7E D4 00087	CLRL	-(SP)	:	
	66		55 DD 00089	PUSHL	R5	:	
		03	FB 0008B	CALLS	#3, LIB\$SIGNAL	:	
		04	0008E	RET		:	1018

; Routine Size: 143 bytes, Routine Base: \$CODE\$ + 0429

```
1030 1019 1 %SBTTL 'NCP$SENDMSG Send a Message to NML'
1031 1020 1 GLOBAL ROUTINE NCP$SENDMSG (LCB, LEN, BFR) :NOVALUE = !
1032 1021 1
1033 1022 1 ++
1034 1023 1 FUNCTIONAL DESCRIPTION:
1035 1024 1
1036 1025 1 This routine sends a message to the NML object over the link
1037 1026 1 described by the LCB argument. The buffer is described by the
1038 1027 1 remaining arguments. System service and IO errors are signalled.
1039 1028 1
1040 1029 1 FORMAL PARAMETERS:
1041 1030 1
1042 1031 1 LCB Address of the link control block
1043 1032 1 LEN Value of the length of the message
1044 1033 1 BFR Address of the message buffer
1045 1034 1
1046 1035 1 IMPLICIT INPUTS:
1047 1036 1
1048 1037 1 NONE
1049 1038 1
1050 1039 1 IMPLICIT OUTPUTS:
1051 1040 1
1052 1041 1 NONE
1053 1042 1
1054 1043 1 ROUTINE VALUE:
1055 1044 1 COMPLETION CODES:
1056 1045 1
1057 1046 1 NONE
1058 1047 1
1059 1048 1 SIDE EFFECTS:
1060 1049 1
1061 1050 1 NONE
1062 1051 1
1063 1052 1 --
1064 1053 1
1065 1054 2 BEGIN
1066 1055 2
1067 1056 2 MAP
1068 1057 2 LCB : REF BBLOCK ! Link control block
1069 1058 2 ;
1070 1059 2
1071 1060 2 LOCAL
1072 1061 2 STATUS, ! Service status
1073 1062 2 IOSB : BBLOCK [8] ! IO status block
1074 1063 2 ;
1075 1064 2
1076 1065 2 EXTERNAL LITERAL
1077 1066 2 NCP$_NETIO ! Network comm error
1078 1067 2 ;
1079 1068 2
1080 1069 2 IF NOT .LCB [LCB$B_STS] ! If link is not open
1081 1070 2 THEN
1082 1071 2 NCP$OPENLINK (.LCB); ! Open the link to executor
1083 1072 2
1084 1073 2 IF CH$RCHAR(.LCB [LCB$L_NCBPTR]) EQL ':' ! If talking to sharable NML,
1085 1074 2 THEN
1086 1075 2 BEGIN
```

2C

20

65

65

6F

PPPPPP

				.EXTRN	NCP\$_NETIO		
			0004	00000	.ENTRY	NCP\$SENDMSG, Save R2	: 1020
	5E		18	C2 00002	SUBL2	#24, SP	: 1069
	52	04	AC	D0 00005	MOVL	LCB, R2	: 1071
	07		62	E8 00009	BLBS	(R2), 1\$	: 1073
			52	DD 0000C	PUSHL	R2	: 1082
FD81	CF		01	FB 0000E	CALLS	#1, NCP\$OPENLINK	: 1085
	3A	0E	B2	91 00013	CMPB	@14(R2), #58	: 1086
			3A	12 00017	BNEQ	4\$	: 1082
	50	00000000'	00	9E 00019	MOVAB	NML RESP QUEUE, R0	: 1085
	6E	00	B0	0F 00020	REMQUE	@0(R0), ENTRY	: 1086
			17	1D 00024	BVS	3\$	: 1082
	50		6E	D0 00026	MOVL	ENTRY, R0	: 1085
08	A0		0C	C1 00029	ADDL3	#12, 8(R0), LENGTH	: 1086
			5E	DD 0002F	PUSHL	SP	: 1082
		08	AE	9F 00031	PUSHAB	LENGTH	: 1085
00000000G	00		02	FB 00034	CALLS	#2, LIB\$FREE_VM	: 1086
			DC	11 00038	BRB	2\$	: 1082
08	AE	08	AC	7D 0003D	MOVQ	LEN, MSGDESC	: 1089
		00000000V	00	9F 00042	PUSHAB	STORE_RESPONSE	: 1091

NCPNETIO
V04-000

Network I/O Routines
NCP\$SENDMSG Send a Message to NML

E 2
15-Sep-1984 23:46:44
14-Sep-1984 12:48:14

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[NCP.SRC]NCPNETIO.B32;1
Page 34
(12)

NCP
V04

00000000G	00	0C	AE 9F 00048	PUSHAB MSGDESC	:	
			02 FB 0004B	CALLS #2, NML\$PROCESS_NICE	:	
			04 00052	RET	:	1075
			7E 7C 00053 4\$:	CLRQ -(SP)	:	1103
			7E 7C 00055	CLRQ -(SP)	:	
		08	AC DD 00057	PUSHL LEN	:	
		0C	AC DD 0005A	PUSHL BFR	:	
			7E 7C 0005D	CLRQ -(SP)	:	
		30	AE 9F 0005F	PUSHAB IOSB	:	
			30 DD 00062	PUSHL #48	:	
	50	04	AC D0 00064	MOVL LCB, R0	:	
	7E	02	A0 3C 00068	MOVZWL 2(R0), -(SP)	:	
			7E D4 0006C	CLRL -(SP)	:	
00000000G	00		0C FB 0006E	CALLS #12, SYSSQIOW	:	
		10	AE 9F 00075	PUSHAB IOSB	:	1104
			50 DD 00078	PUSHL STATUS	:	
			8F DD 0007A	PUSHL #NCP\$ NETIO	:	
FEBB	CF		03 FB 00080	CALLS #3, NCP\$SIGNETERR	:	
			04 00085	RET	:	1108

; Routine Size: 134 bytes, Routine Base: \$CODE\$ + 04B8

```
: 1121 1109 1 %SBTTL 'STORE_RESPONSE Store a response from sharable NML'
: 1122 1110 1 ROUTINE store_response (resp_desc): NOVALUE =
: 1123 1111 1
: 1124 1112 1 ++
: 1125 1113 1
: 1126 1114 1 This routine is called by NML$PROCESS_NICE for each response
: 1127 1115 1 that it generates as a result of processing a single NICE message.
: 1128 1116 1 All we do is store the response messages away in a queue in the
: 1129 1117 1 order in which they were generated, and de-queue them later when
: 1130 1118 1 we wish to 'read' a response.
: 1131 1119 1
: 1132 1120 1 Inputs:
: 1133 1121 1
: 1134 1122 1 resp_desc = Address of descriptor of NICE response message
: 1135 1123 1
: 1136 1124 1 Outputs:
: 1137 1125 1
: 1138 1126 1 None
: 1139 1127 1 --
: 1140 1128 1
: 1141 1129 2 BEGIN
: 1142 1130 2
: 1143 1131 2 BUILTIN INSQUE;
: 1144 1132 2
: 1145 1133 2 MAP
: 1146 1134 2 resp_desc: REF BBLOCK; ! Address of response descriptor
: 1147 1135 2
: 1148 1136 2 LOCAL
: 1149 1137 2 status,
: 1150 1138 2 length, ! Length of block containing response
: 1151 1139 2 entry: REF VECTOR; ! Address of block to contain response
: 1152 1140 2
: 1153 1141 2 length = .resp_desc [dsc$w_length] + 12; ! Add response length + overhead
: 1154 1142 2
: 1155 1143 2 status = LIB$GET_VM(length, entry); ! Allocate dynamic memory
: 1156 1144 2
: 1157 1145 2 IF NOT .status ! If error detected,
: 1158 1146 2 THEN ! then signal fatal error
: 1159 1147 2 SIGNAL_STOP(.status);
: 1160 1148 2
: 1161 1149 2 entry [2] = .resp_desc [dsc$w_length]; ! Store length of response message
: 1162 1150 2 CH$MOVE(.resp_desc [dsc$w_length], ! Copy message to new block
: 1163 1151 2 .resp_desc [dsc$a_pointer],
: 1164 1152 2 entry [3]);
: 1165 1153 2
: 1166 1154 2 INSQUE(.entry, .nml_resp_queue [1]); ! Insert at end of queue
: 1167 1155 2
: 1168 1156 1 END;
```

007C 000C0 STORE_RESPONSE:

5E	08	C2	00002	WORD	Save R2,R3,R4,R5,R6	: 1110
52	04	AC	D0 00005	SUBL2	#8, SP	: 1141
				MOVL	RESP_DESC, R2	

NCPNETIO
V04-000

Network I/O Routines
STORE_RESPONSE Store a response from sharable

G 2
15-Sep-1984 23:46:44
14-Sep-1984 12:48:14

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[NCP.SRC]NCPNETIO.B32;1
Page 36
(13)

04	AE	62	3C	00009	MOVZWL	(R2), LENGTH	:	
04	AE	0C	C0	0000D	ADDL2	#12, LENGTH	:	
		5E	DD	00011	PUSHL	SP	1143	
		08	AE	9F	00013	PUSHAB	LENGTH	:
00000000G	00	02	FB	00016	CALLS	#2, LIB\$GET_VM	:	
	09	50	E8	0001D	BLBS	STATUS, 1\$	1145	
		50	DD	00020	PUSHL	STATUS	1147	
00000000G	00	01	FB	00022	CALLS	#1, LIB\$STOP	:	
	56	6E	D0	00029	1\$:	MOVL	ENTRY, R6	1149
	08	62	3C	0002C	MOVZWL	(R2), 8(R6)	:	
0C	A6	62	28	00030	MOVZWL	(R2), 8(R6)	1152	
		00	9E	00036	MOVZWL	(R2), 8(R6)	:	
		50	00000000'	00	9E	00036	1154	
		00	B0	66	0E	0003D	:	
				04	00041	RET	1156	

; Routine Size: 66 bytes, Routine Base: \$CODE\$ + 053E


```
: 1170 1157 1 %SBTTL 'NCP$READRSP Read and Decode an NML Response'
: 1171 1158 1 GLOBAL ROUTINE NCP$READRSP (LCB, LEN, BFR, SHO) =
: 1172 1159 1
: 1173 1160 1 ++
: 1174 1161 1 FUNCTIONAL DESCRIPTION:
: 1175 1162 1
: 1176 1163 1 This routine reads a message from NML and decodes it.
: 1177 1164 1 If the message is an error response, the error is signaled and
: 1178 1165 1 control does not return to the caller.
: 1179 1166 1 If the message is a data return or a done status, the message is
: 1180 1167 1 returned via LEN, BFR and the first byte is returned as the value of
: 1181 1168 1 the routine. LEN and BFR form a descriptor of the data beyond the
: 1182 1169 1 error status byte, detail and error message. If the error status
: 1183 1170 1 is SUC, DON or MOR, and there is a detail or error message, an
: 1184 1171 1 error is signaled to print these but control returns normally to
: 1185 1172 1 the caller.
: 1186 1173 1
: 1187 1174 1 If an error contains data, it is assumed to be an entity for the
: 1188 1175 1 error and the entity code is formatted and included in the error
: 1189 1176 1 message. Entity codes may also occur with success codes and in
: 1190 1177 1 this case the data is printed as an entity if the message is not
: 1191 1178 1 a show or list command, indicated by the SHO parameter.
: 1192 1179 1
: 1193 1180 1 FORMAL PARAMETERS:
: 1194 1181 1
: 1195 1182 1 LCB Address of link control block
: 1196 1183 1 LEN Address for return of length of buffer
: 1197 1184 1 BFR Address for return of address of buffer
: 1198 1185 1 SHO True if the command is show or list
: 1199 1186 1
: 1200 1187 1 IMPLICIT INPUTS:
: 1201 1188 1
: 1202 1189 1 NCP$GL_ENTITY Entity number sent in original message
: 1203 1190 1 (If negative, then system-specific entity)
: 1204 1191 1
: 1205 1192 1 IMPLICIT OUTPUTS:
: 1206 1193 1
: 1207 1194 1 NONE
: 1208 1195 1
: 1209 1196 1 ROUTINE VALUE:
: 1210 1197 1 COMPLETION CODES:
: 1211 1198 1
: 1212 1199 1 Value of first byte of message, or error signalled
: 1213 1200 1
: 1214 1201 1 SIDE EFFECTS:
: 1215 1202 1
: 1216 1203 1 NONE
: 1217 1204 1
: 1218 1205 1 --
: 1219 1206 1
: 1220 1207 1 BEGIN
: 1221 1208 1
: 1222 1209 1 MAP
: 1223 1210 2 LCB : REF BBLOCK ! Link control block
: 1224 1211 2 ;
: 1225 1212 2
: 1226 1213 2 LITERAL
```

```
: 1227      1214 2      RSPSIZ = 32,      ! Size of response buffer required
: 1228      1215 2      DTLSIZ = 32,      ! Size of detail buffer required
: 1229      1216 2      ENTSIZ = 32      ! Size of entity code buffer
: 1230      1217 2      ;
: 1231      1218 2
: 1232      1219 2      LOCAL
: 1233      1220 2      STATUS,      ! Service status return
: 1234      1221 2      OUTLEN,      ! Length in a buffer
: 1235      1222 2      IOSB : BBLOCK [8],      ! QIO status
: 1236      1223 2      CTR,      ! General temps
: 1237      1224 2      PTR,
: 1238      1225 2      CODE,
: 1239      1226 2      ENTIFY,      ! Entity number (negative if sys-specific)
: 1240      1227 2      RSP,      ! Pointer for response text
: 1241      1228 2      COMMA,      ! Pointer to separator before detail
: 1242      1229 2      DTL,      ! Pointer for detail text
: 1243      1230 2      ERR,      ! Pointer for error text
: 1244      1231 2      ENT,      ! Pointer for entity code text
: 1245      1232 2      IDX,      ! Index into tables
: 1246      1233 2      JUNK,      ! Throw away temporary
: 1247      1234 2      DETAIL,      ! Value of detail word
: 1248      1235 2      DTLTBL,      ! Address of detail table
: 1249      1236 2      ;
: 1250      1237 2
: 1251      1238 2      OWN
: 1252      1239 2      DTLBUF : VECTOR [DTLSIZ, BYTE], ! Detail bu
: 1253      1240 2      RSPBUF : VECTOR [RSPSIZ, BYTE], ! Response
: 1254      1241 2      ENTDESC : VECTOR [2], ! Descriptor for string
: 1255      1242 2      ENTBUF : VECTOR [ENTSIZ, BYTE] ! Entity string buffer
: 1256      1243 2      ;
: 1257      1244 2
: 1258      1245 2
: 1259      1246 2      EXTERNAL LITERAL
: 1260      1247 2      NCP$_NMLRSP,      ! NML response message
: 1261      1248 2      NCP$_NETIO      ! Network communication error
: 1262      1249 2      ;
: 1263      1250 2
: 1264      1251 2      EXTERNAL
: 1265      1252 2      NCP$GA_TBL_NMLSTS,      ! NML status return codes
: 1266      1253 2      NCP$GA_TBL_FOPDTL,      ! File operations detail codes
: 1267      1254 2      NCP$GA_TBL_NCEDTL,      ! Network communications detail codes
: 1268      1255 2      NCP$GA_TBL_VMSENTDTL,      ! Detail table of VMS specific entities
: 1269      1256 2      NCP$GA_TBL_ENTDTL,      ! Detail table of entities
: 1270      1257 2      NCP$GA_TBL_OPEDTL;      ! Detail table of operation failures
: 1271      1258 2
: 1272      1259 2      EXTERNAL ROUTINE
: 1273      1260 2      NCP$FAOSET : NOVALUE,      ! Setup to convert entity
: 1274      1261 2      NCP$SHOENTITY : NOVALUE,      ! Convert entity
: 1275      1262 2      NCP$FAOL : NOVALUE      ! Convert fao string for entity
: 1276      1263 2      ;
: 1277      1264 2
: 1278      1265 2      .LEN = 0;      ! Set callers data
: 1279      1266 2      .BFR = NCP$GT_RSPBFR;
: 1280      1267 2
: 1281      1268 2      IF CH$RCHAR(.LCB [LCB$_NCBPTR]) EQL ':'      ! If talking to sharable NML.
: 1282      1269 2      THEN
: 1283      1270 2      BEGIN
```



```
1284 1271 3      BUILTIN REMQUE;
1285 1272 3      LOCAL
1286 1273 3          length,
1287 1274 3          entry: REF VECTOR;
1288 1275 3      IF REMQUE(.nml_resp_queue [0], entry) ! De-queue next one. If none,
1289 1276 3      THEN
1290 1277 3          SIGNAL_STOP(NCP$_NETIO, SS$_ABORT); ! signal fatal error
1291 1278 3      ctr = .entry [2]; ! Copy length of response
1292 1279 3      ptr = ncp$gt_rspbfr; ! Set address of buffer
1293 1280 3      CH$MOVE(.ctr, entry [3], .ptr); ! Copy response into buffer
1294 1281 3      length = .ctr + 12; ! Set length of container block
1295 1282 3      LIB$FREE_VM(length, entry); ! Deallocate container block
1296 1283 3      END
1297 1284 3  ELSE ! Else, read response from logical link
1298 1285 3      BEGIN
1299 1286 3          STATUS = $QIOW ! Read the message from NML
1300 1287 3          (
1301 1288 3              CHAN = .LCB [LCB$W_CHAN],
1302 1289 3              FUNC = IOS$READVBLR,
1303 1290 3              IOSB = IOSB,
1304 1291 3              P1 = NCP$GT_RSPBFR,
1305 1292 3              P2 = NCP$C_RSPSIZ
1306 1293 3          );
1307 1294 3      NCP$SIGNETERR (NCP$_NETIO, .STATUS, IOSB);
1308 1295 3
1309 1296 3      CTR = .IOSB [0, 16, 16, 0]; ! Point and count into message
1310 1297 3      PTR = NCP$GT_RSPBFR;
1311 1298 3      END;
1312 1299 3
1313 1300 3      :
1314 1301 3      We need to set some defaults in case the message is bad
1315 1302 3
1316 1303 3      RSP = UPLIT (%ASCIC 'unrecognized'); ! Some default text for message
1317 1304 3      COMMA = UPLIT (%ASCIC '');
1318 1305 3      DTL = UPLIT (%ASCIC '');
1319 1306 3      ENT = UPLIT (%ASCIC '');
1320 1307 3      ERR = UPLIT (%ASCIC '');
1321 1308 3
1322 1309 3      IF .CTR EQL 0 ! If message is short, signal now
1323 1310 3      THEN
1324 1311 3          SIGNAL_STOP (NCP$_NMLRSP, 5, .RSP, .COMMA, .DTL, .ENT, .ERR)
1325 1312 3      ;
1326 1313 3
1327 1314 3      CODE = .(.PTR) <0, 8, 1>; ! First byte is a code
1328 1315 3
1329 1316 3      IF NOT NCP$TABLESEARCH ! Find the code text if possible
1330 1317 3      (
1331 1318 3          .CODE <0, 8, 0> ! Code byte
1332 1319 3          NCP$GA_TBL_NMLSTS, ! Table
1333 1320 3          RSP ! Return address of counted string
1334 1321 3      )
1335 1322 3
1336 1323 3      THEN
1337 1324 3      BEGIN
1338 1325 3          $FAO ! If not found, make some text
1339 1326 3      (
1340 1327 3
```

```
: 1341      ASCID ('management return # !SB'),
: 1342      OUTLEN,
: 1343      UPLIT (RSPSIZ-1, RSPBUF+1),
: 1344      .CODE
: 1345      );
: 1346      RSPBUF [0] = .OUTLEN;      ! As a counted string
: 1347      RSP = RSPBUF              ! Point to it
: 1348      END
: 1349      ;
: 1350      ;
: 1351      DETAIL = -1;                ! No detail yet
: 1352      ;
: 1353      IF .CTR GEQ 3              ! Is there a detail word
: 1354      THEN
: 1355          BEGIN
: 1356              DETAIL = .(.PTR+1) <0, 16, 1>; ! Obtain the word
: 1357              IF .DETAIL NEQ -1          ! Ignore value?
: 1358              THEN
: 1359                  BEGIN
: 1360                      DTLTBL =
: 1361                      BEGIN
: 1362                          SELECTONE .CODE OF
: 1363                          SET
: 1364                          [NMA$C_STS_FOP, NMA$C_STS_FIO, NMA$C_STS_FCO] :
: 1365                          NCP$GA_TBL_FOPDTL ! File io errors
: 1366                          ;
: 1367                          [NMA$C_STS_MLD, NMA$C_STS_MCF] :
: 1368                          NCP$GA_TBL_NCEDTL ! Network io errors
: 1369                          ;
: 1370                          [NMA$C_STS_OPE] :
: 1371                          NCP$GA_TBL_OPEDTL ! Operation failure
: 1372                          ;
: 1373                          [NMA$C_STS_CMP, NMA$C_STS_IDE, NMA$C_STS_STA] :
: 1374                          ! Errors with entities
: 1375                          IF .NCP$GL_ENTITY LSS 0 ! If system-specific entity
: 1376                          THEN
: 1377                              NCP$GA_TBL_VMSENTDTL ! VMS entities
: 1378                              ;
: 1379                              ELSE
: 1380                                  NCP$GA_TBL_ENTDTL; ! DNA entities
: 1381                                  [OTHERWISE] : ! Details not valid
: 1382                                  BEGIN
: 1383                                      IF .DETAIL EQL 0 ! Zero is null detail here
: 1384                                      THEN 1 ! Null detail if not valid
: 1385                                      ELSE 0 ! But report non zero detail
: 1386                                      END
: 1387                                  ;
: 1388                              TES
: 1389                              END
: 1390                          ;
: 1391                          IF .CODE EQL NMA$C_STS_OPE ! If operation failure
: 1392                          AND ! and entity is line
: 1393                          (.NCP$GL_ENTITY EQL
: 1394                          NMA$C_ENT_LIN
: 1395                          OR
: 1396                          .NCP$GL_ENTITY EQL
: 1397                          NMA$C_ENT_CIR)
```

```
1398 1385 5
1399 1386 4
1400 1387 5
1401 1388 5
1402 1389 5
1403 1390 5
1404 1391 5
1405 1392 5
1406 1393 5
1407 1394 5
1408 1395 5
1409 1396 5
1410 1397 5
1411 1398 5
1412 1399 5
1413 1400 5
1414 1401 5
1415 1402 5
1416 1403 5
1417 1404 5
1418 1405 5
1419 1406 5
1420 1407 6
1421 1408 6
1422 1409 5
1423 1410 5
1424 1411 5
1425 1412 5
1426 1413 5
1427 1414 5
1428 1415 5
1429 1416 5
1430 1417 5
1431 1418 5
1432 1419 5
1433 1420 5
1434 1421 6
1435 1422 6
1436 1423 6
1437 1424 6
1438 1425 6
1439 1426 6
1440 1427 6
1441 1428 6
1442 1429 7
1443 1430 7
1444 1431 7
1445 1432 7
1446 1433 7
1447 1434 7
1448 1435 7
1449 1436 7
1450 1437 7
1451 1438 7
1452 1439 7
1453 1440 7
1454 1441 7

      THEN
      BEGIN
        LOCAL
          PREBUF : VECTOR [40, BYTE],      ! Buffer for string to proceed
                                              ! each detail message.
          PRELEN,                          ! Length of string to proceed
                                              ! each detail message.
          LOCPTR;                          ! Local pointer

          LOCPTR = PREBUF;                  ! Init pointer into buffer

      Build the string which will precede the detail text so that each detail
      string output will line-up under the error text.  For example:

          %facility-L-ident, error text      ! Original error message
          %facility-L-ident, error text<CR><LF> ! Message with two detail
          < SPACES >, detail text<CR><LF>    strings appended.
          < SPACES >, detail text

          PRELEN = ( CH$FIND.CH(.(.PTR+3), ! Get the number of characters
                      .PTR + 4, %C' ') ) ! in the facility and ident
                      - (.PTR + 4);      ! portion of error message

          (.LOCPTR) <0, 16> = %X'0A0D';    ! Store <CR><LF> in buffer,
          LOCPTR = CH$FILL( %C' ', PRELEN, ! .LOCPTR + 2 ); ! some spaces,
          (.LOCPTR) <0, 16> = %ASCII', ' ; ! and a " "
          PRELEN = .PRELEN + 4;            ! Length = length of facility,
                                              ! text plus <CR><LF> and " "

          LOCPTR = .PTR + 4 +              ! Point to end of original
                      .(.PTR + 3) < 0, 8 >; ! error message text.

          INCR INDEX FROM 0 TO 16 DO
          BEGIN
            IF .DETAIL < .INDEX, 1, 0 > ! If status or error bit is set,
            AND                          ! and it's in the table,
            NCP$TABLESEARCH (.INDEX, .DTL, DTL)
            AND                          ! and there's room in the
            .PRELEN + .(.DTL) < 0, 8 > ! response buffer.
            LEQ .PTR + NCP$C_RSPSI2 - .LOCPTR
            THEN
            BEGIN
              LOCPTR = CH$MOVE
              (
                .PRELEN,
                PREBUF,
                .LOCPTR
              );
              ! Append the string which
              ! preceeds each detail message
              ! to the end of the error
              ! message

              LOCPTR = CH$MOVE
              (
                .(.DTL) <0,8>,
                .DTL + 1,
              );
              ! Append detail to end of the
              ! error message
```

```

1455      1442 7      .LOCPTR
1456      1443 7      );
1457      1444 6
1458      1445 5      END;
1459      1446 5
1460      1447 5      (.PTR + 3) < 0, 8 > =
1461      1448 5      .LOCPTR - .PTR - 4;
1462      1449 5      CTR < 0, 8 > = .LOCPTR - .PTR;
1463      1450 5      DTLTBL = 1;
1464      1451 5      DTLBUF [0] = 0;
1465      1452 5      DTL = DTLBUF;
1466      1453 5
1467      1454 5      END
1468      1455 5
1469      1456 4      ELSE
1470      1457 4      IF .CODE EQL NMASC_STS_PVA
1471      1458 4      OR
1472      1459 4      .CODE EQL NMASC_STS_PLO
1473      1460 4      OR
1474      1461 4      .CODE EQL NMASC_STS_PNA
1475      1462 4      OR
1476      1463 4      .CODE EQL NMASC_STS_PTY
1477      1464 4      OR
1478      1465 4      .CODE EQL NMASC_STS_PGP
1479      1466 4      OR
1480      1467 4      .CODE EQL NMASC_STS_PMS
1481      1468 4      THEN
1482      1469 5      BEGIN
1483      1470 5      NCP$FORMATPARM
1484      1471 5      (
1485      1472 5      .NCP$GL_ENTITY,
1486      1473 5      DETAIL,
1487      1474 5      TRUE,
1488      1475 5      FALSE,
1489      1476 5      UPLIT (DTLSIZ - 1, DTLBUF + 1),
1490      1477 5      OUTLEN,
1491      1478 5      JUNK
1492      1479 5      );
1493      1480 5      DTLBUF [0] = .OUTLEN;
1494      1481 5      DTL = DTLBUF;
1495      1482 5      DTLTBL = 1
1496      1483 5      END
1497      1484 4      ;
1498      1485 4
1499      1486 4      IF .DTLTBL NEQ 1
1500      1487 4      AND
1501      1488 5      (
1502      1489 5      .DTLTBL EQL 0
1503      1490 5      OR
1504      1491 6      (
1505      1492 6      IF .DTLTBL NEQ 0
1506      1493 6      THEN
1507      1494 6      NOT NCP$TABLESEARCH (.DETAIL, .DTLTBL, DTL)
1508      1495 6      ELSE
1509      1496 6      TRUE
1510      1497 6      )
1511      1498 5      )

```

```

: 1512      1499 4      THEN
: 1513      1500 5      BEGIN
: 1514      1501 5      $FAO      ! Put out in some standard way
: 1515      1502 5      (
: 1516      1503 5      ASCID ('detail # !UW'),
: 1517      1504 5      OUTLEN,
: 1518      1505 5      UPLIT (DTLSIZ-1, DTLBUF+1),
: 1519      1506 5      .DETAIL
: 1520      1507 5      );
: 1521      1508 5      DTLBUF [0] = .OUTLEN;      ! As counted string
: 1522      1509 5      DTL = DTLBUF
: 1523      1510 5      END
: 1524      1511 4      END
: 1525      1512 3      END
: 1526      1513 2      ;
: 1527      1514 2      IF .CTR GEQU 4      ! If there is enough for system
: 1528      1515 2      THEN      ! Specific error text
: 1529      1516 2      BEGIN
: 1530      1517 3      IF .CTR GEQU (4 + .(.PTR+3) <0, 8, 0> )
: 1531      1518 4      THEN      ! And the text is valid
: 1532      1519 3      BEGIN
: 1533      1520 4      ERR = .PTR + 3;      ! Point to the counted string
: 1534      1521 4      .LEN = .CTR - (.(.PTR+3) <0, 8, 0>) - 4;      ! Adjust returned length
: 1535      1522 4      .BFR = ..BFR + 4 + (.(.PTR+3) <0, 8, 0>)      ! And buffer beyond it
: 1536      1523 5      END
: 1537      1524 4      ELSE
: 1538      1525 3      ERR = UPLIT (%ASCIC '%NCP-W-ERRRSP, invalid error text in listener response')
: 1539      1526 3      END
: 1540      1527 3      ;
: 1541      1528 2      ;
: 1542      1529 2      ;
: 1543      1530 2      ;
: 1544      1531 2      ;
: 1545      1532 2      ;
: 1546      1533 2      ;
: 1547      1534 2      ;
: 1548      1535 2      IF ..LEN NEQ 0      ! Is there an entity for the message
: 1549      1536 2      AND
: 1550      1537 2      NOT .SHO      ! and this is not a show or list
: 1551      1538 2      THEN
: 1552      1539 3      BEGIN
: 1553      1540 3      .LEN = 0;      ! Return no data to caller
: 1554      1541 3      ENTDSC [0] = ENTSIZ - 1;      ! Descriptor for output is buffer
: 1555      1542 3      ENTDSC [1] = ENTBUF + 1;      ! Less one byte for count
: 1556      1543 3      ENT = ENTBUF;      ! Set counted string address
: 1557      1544 3      IF .NCP$GL_FNC_CODE NEQ NMA$C_FNC_TES      ! Loop return with test data
: 1558      1545 3      THEN
: 1559      1546 4      BEGIN
: 1560      1547 4      PTR = ..BFR;      ! Set pointer to entity code
: 1561      1548 4      NCP$FAOSET ();      ! Setup conversion routines
: 1562      1549 4      NCP$SHOENTITY (PTR);      ! Convert to fao parameters
: 1563      1550 4      NCP$FAOL (ENTDSC);      ! Convert to text
: 1564      1551 4      ENTBUF [0] = .ENTDSC [0];      ! Make counted string
: 1565      1552 4      END
: 1566      1553 3      ELSE
: 1567      1554 4      BEGIN
: 1568      1555 4      $FAO      ! Convert test data if loop return
```

```

: 1569 P 1556 4 (
: 1570 P 1557 4 (
: 1571 P 1558 4 IF .CODE EQL NMASC_STS_PVA ! Special case the text for
: 1572 P 1559 4 THEN ASCID ('Maximum data length = !UW') ! a loop message
: 1573 P 1560 4 ELSE ASCID ('Messages not looped = !UW')
: 1574 P 1561 4 )
: 1575 P 1562 4 )
: 1576 P 1563 4 OUTLEN,
: 1577 P 1564 4 ENTDS, ! Descriptor of buffer
: 1578 P 1565 4 )..BFR ! Stack the data (word of loop count)
: 1579 1566 4 ENTBUF [0] = .OUTLEN ! Set counter for this message
: 1580 1567 4 END
: 1581 1568 3 END
: 1582 1569 2 ;
: 1583 1570 2 ;
: 1584 1571 2 IF CH$RCHAR(.DTL) NEQ 0 ! If text following message,
: 1585 1572 2 THEN ! then delimit with a comma
: 1586 1573 2 COMMA = UPLIT(%ASCIC ',');
: 1587 1574 2 IF
: 1588 1575 2 (
: 1589 1576 3 (
: 1590 1577 4 (.CODE NEQ NMASC_STS_MOR ! If a not a success code
: 1591 1578 4 AND
: 1592 1579 4 .CODE NEQ NMASC_STS_SUC
: 1593 1580 4 AND
: 1594 1581 4 .CODE NEQ NMASC_STS_DON
: 1595 1582 4 AND
: 1596 1583 4 .CODE NEQ NMASC_STS_PAR
: 1597 1584 4 )
: 1598 1585 4 )
: 1599 1586 3 AND
: 1600 1587 3 CH$RCHAR (.RSP) NEQ 0 ! and the response message is here
: 1601 1588 3 )
: 1602 1589 2 OR
: 1603 1590 2 CH$RCHAR (.DTL) NEQ 0 ! or any of the text strings are here
: 1604 1591 2 OR
: 1605 1592 2 CH$RCHAR (.ERR) NEQ 0 ! then print the error
: 1606 1593 2 THEN
: 1607 1594 2 SIGNAL (NCP$_NMLRSP, 5, .RSP, .COMMA, .DTL, .ENT, .ERR)
: 1608 1595 2 ;
: 1609 1596 2 RETURN .CODE ! Return data to caller
: 1610 1597 2
: 1611 1598 2
: 1612 1599 1 END;
```

```

                                .PSECT $PLITS,NOWRT,NOEXE,2
00 00 64 65 7A 69 6E 67 6F 63 65 72 6E 75 0C 0005B .BLKB 1
                                0005C P.AAL: .ASCII <12>\unrecognized\<0><0><0>
                                0006B
                                00 00 00 00 0006C P.AAM: .ASCII <0><0><0><0>
                                00 00 00 00 00070 P.AAN: .ASCII <0><0><0><0>
                                00 00 00 00 00074 P.AAO: .ASCII <0><0><0><0>
                                00 00 00 00 00078 P.AAP: .ASCII <0><0><0><0>
75 74 65 72 20 74 6E 65 6D 65 67 61 6E 61 6D 0007C P.AAR: .ASCII \management return # !SB\<0>
```



```
00 42 53 21 20 23 20 6E 72 0008B
00000017 00094 P.AAQ: .LONG 23
00000000 00098 .ADDRESS P.AAR
0000001F 0009C P.AAS: .LONG 31
00000000 000A0 .ADDRESS RSPBUF+1
0000001F 000A4 P.AAT: .LONG 31
00000000 000A8 .ADDRESS DTLBUF+1
57 55 21 20 23 20 6C 69 61 74 65 64 000AC P.AAV: .ASCII \detail # !UW\
0000000C 000B8 P.AAU: .LONG 12
00000000 000BC .ADDRESS P.AAV
0000001F 000C0 P.AAW: .LONG 31
00000000 000C4 .ADDRESS DTLBUF+1
2C 50 53 52 52 52 45 2D 57 2D 50 43 4E 25 36 000C8 P.AAX: .ASCII \6%NCP-W-ERRRSP, invalid error text in li\
20 72 6F 72 72 65 20 64 69 6C 61 76 6E 69 20 000D7
65 73 6E 6F 70 73 65 72 20 72 65 6E 65 74 73 000E6
. ASCII \stener response\<0>
65 6C 20 61 74 61 64 20 6D 75 6D 69 78 61 4D 00100 P.AAZ: .ASCII \Maximum data length = !UW\<0><0><0>
00 00 00 57 55 21 20 3D 20 68 74 67 6E 0010F
00000019 0011C P.AAY: .LONG 25
00000000 00120 .ADDRESS P.AAZ
6F 6C 20 74 6F 6E 20 73 65 67 61 73 73 65 4D 00124 P.ABB: .ASCII \Messages not looped = !UW\<0><0><0>
00 00 00 57 55 21 20 3D 20 64 65 70 6F 00133
00000019 00140 P.ABA: .LONG 25
00000000 00144 .ADDRESS P.ABB
00 00 2C 01 00148 P.ABC: .ASCII <1>\,\<0><0>
. PSECT $OWNS$,NOEXE,2
0007C DTLBUF: .BLKB 32
0009C RSPBUF: .BLKB 32
000BC ENTDCS: .BLKB 8
000C4 ENTBUF: .BLKB 32
. EXTRN NCP$ NMLRSP, NCP$GA_TBL_NMLSTS
. EXTRN NCP$GA_TBL_FOPDTL
. EXTRN NCP$GA_TBL_NCEDTL
. EXTRN NCP$GA_TBL_VMSENTDTL
. EXTRN NCP$GA_TBL_ENTDTL
. EXTRN NCP$GA_TBL_OPEDTL
. EXTRN NCP$FAOSET, NCP$SHOENTITY
. EXTRN NCP$FAOL, SYSSFAO
. PSECT $CODE$,NOWRT,2
OFFC 00000
. ENTRY NCP$READRSP, Save R2,R3,R4,R5,R6,R7,R8,R9,- : 1158
R10,R11
MOVAB -92(SP), SP
CLRL @LEN : 1265
MOVAB NCP$GT_RSPBFR, @BFR : 1266
MOVL LCB, R2 : 1268
CMPB @14(R2), #58
BNEQ 2$
MOVAB NML_RESP_QUEUE, R0 : 1275
REMQUE @0(R0), ENTRY
BVC 1$
PUSHL #44 : 1277
```

NCP
V04

00

75

2C
20

65

65

6F

		03		57	D1	00132		CMPL	CTR, #3		1340
				03	18	00135		BGEQ	7\$		
				01FD	31	00137	6\$:	BRW	27\$		
	1C	AE	01	A8	32	0013A	7\$:	CVTWL	1(R8), DETAIL		1343
FFFFFFFF	8F	1C		AE	D1	0013F		CMPL	DETAIL, #-1		1344
				EE	13	00147		BEQL	6\$		
FFFFFFEE	8F			5A	D1	00149		CMPL	CODE, #-18		1351
				12	13	00150		BEQL	8\$		
FFFFFFF2	8F			5A	D1	00152		CMPL	CODE, #-14		
				12	19	00159		BLSS	9\$		
FFFFFFF3	8F			5A	D1	0015B		CMPL	CODE, #-13		
				09	14	00162		BGTR	9\$		
59 00000000G				00	9E	00164	8\$:	MOVAB	NCP\$GA_TBL_FOPDTL, DTLTBL		
				6E	11	0016B		BRB	17\$		
FFFFFFEB	8F			5A	D1	0016D	9\$:	CMPL	CODE, #-21		1354
				09	13	00174		BEQL	10\$		
FFFFFFED	8F			5A	D1	00176		CMPL	CODE, #-19		
				09	12	0017D		BNEQ	11\$		
59 00000000G				00	9E	0017F	10\$:	MOVAB	NCP\$GA_TBL_NCEDTL, DTLTBL		
				53	11	00186		BRB	17\$		
FFFFFFE7	8F			5A	D1	00188	11\$:	CMPL	CODE, #-25		1357
				09	12	0018F		BNEQ	12\$		
59 00000000G				00	9E	00191		MOVAB	NCP\$GA_TBL_OPEDTL, DTLTBL		
				41	11	00198		BRB	17\$		
FFFFFFF5	8F			5A	D1	0019A	12\$:	CMPL	CODE, #-11		1360
				12	13	001A1		BEQL	13\$		
FFFFFFF7	8F			5A	D1	001A3		CMPL	CODE, #-9		
				23	19	001AA		BLSS	15\$		
FFFFFFF8	8F			5A	D1	001AC		CMPL	CODE, #-8		
				1A	14	001B3		BGTR	15\$		
00000000G				00	D5	001B5	13\$:	TSTL	NCP\$GL_ENTITY		1362
				09	18	001BB		BGEQ	14\$		
59 00000000G				00	9E	001BD		MOVAB	NCP\$GA_TBL_VMSENTDTL, DTLTBL		
				15	11	001C4		BRB	17\$		
59 00000000G				00	9E	001C6	14\$:	MOVAB	NCP\$GA_TBL_ENTDTL, DTLTBL		
				0C	11	001CD		BRB	17\$		
		1C		AE	D5	001CF	15\$:	TSTL	DETAIL		1369
				05	12	001D2		BNEQ	16\$		
59				01	D0	001D4		MOVL	#1, DTLTBL		
				02	11	001D7		BRB	17\$		
				59	D4	001D9	16\$:	CLRL	DTLTBL		
FFFFFFE7	8F			5A	D1	001DB	17\$:	CMPL	CODE, #-25		1378
				0F	12	001E2		BNEQ	18\$		
50 00000000G				00	D0	001E4		MOVL	NCP\$GL_ENTITY, R0		1380
01				50	D1	001EB		CMPL	R0, #1		
				08	13	001EE		BEQL	19\$		
03				50	D1	001F0		CMPL	R0, #3		1383
				03	13	001F3	18\$:	BEQL	19\$		
				0093	31	001F5		BRW	23\$		
				AE	9E	001F8	19\$:	MOVAB	PREBUF, LOCPTR		1395
04 A8 03		2C		2C	3A	001FC		LOCC	#44, 3(R8), 4(R8)		1407
				02	12	00202		BNEQ	20\$		
				51	D4	00204		CLRL	R1		
				50	9E	00206	20\$:	MOVAB	4(R8), R0		1409
56				50	C3	0020A		SUBL3	R0, R1, PRELEN		
				63	8F	0020E		MOVW	#2573, (LOCPTR)		1411
56 20		0A0D		00	2C	00213		MOVCS	#0, (SP), #32, PRELEN, 2(LOCPTR)		1412

			02	A3	00218																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																									
--	--	--	----	----	-------	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

		59	DD	002FF	PUSHL	DTLTBL	:
		AE	D	00301	PUSHL	DETAIL	:
00000000V	00	03	FB	00304	CALLS	#3, NCP\$TABLESEARCH	:
	29	50	E8	0030B	BLBS	RO, 27\$	:
		AE	DD	0030E	PUSHL	DETAIL	1507
		00	9F	00311	PUSHAB	P.AAW	:
		AE	9F	00317	PUSHAB	OUTLEN	:
		00	9F	0031A	PUSHAB	P.AAU	:
00000000G	00	04	FB	00320	CALLS	#4, SYSS\$FAO	:
00000000'	00	AE	90	00327	MOVB	OUTLEN, DTLBUF	1508
	20	00	9E	0032F	MOVAB	DTLBUF, DTL	1509
		57	D1	00337	CMPL	CTR, #4	1515
		32	1F	0033A	BLSSU	29\$	:
		A8	9A	0033C	MOVZBL	3(R8), RO	1518
		04	C0	00340	ADDL2	#4, RO	:
		57	D1	00343	CMPL	CTR, RO	:
		1F	1F	00346	BLSSU	28\$	:
		A8	9E	00348	MOVAB	3(R8), ERR	1521
		A8	9A	0034C	MOVZBL	3(R8), RO	1522
		50	C2	00350	SUBL2	RO, R7	:
08	BC	A7	9E	00353	MOVAB	-4(R7), @LEN	:
		A8	9A	00358	MOVZBL	3(R8), RO	1523
		BC	C0	0035C	ADDL2	@BFR, RO	:
0C	BC	A0	9E	00360	MOVAB	4(R0), @BFR	:
		07	11	00365	BRB	29\$	:
		00	9E	00367	MOVAB	P.AAX, ERR	1526
		BC	D5	0036E	TSTL	@LEN	1535
		58	13	00371	BEQL	30\$	:
		AC	E8	00373	BLBS	SHO, 30\$	1537
		BC	D4	00377	CLRL	@LEN	1540
00000000'	00	1F	D0	0037A	MOVL	#31, ENTDC	1541
00000000'	00	00	9E	00381	MOVAB	ENTBUF+1, ENTDC+4	1542
	04	AE	00	0038C	MOVAB	ENTBUF, ENT	1543
		12	00	00394	CMPL	NCP\$GL_FNC_CODE, #18	1544
		30	13	0039B	BEQL	31\$	:
		BC	D0	0039D	MOVL	@BFR, PTR	1547
00000000G	00	00	FB	003A2	CALLS	#0, NCP\$FAOSET	1548
		AE	9F	003A9	PUSHAB	PTR	1549
00000000G	00	01	FB	003AC	CALLS	#1, NCP\$SHOENTITY	:
		00	9F	003B3	PUSHAB	ENTDC	1550
00000000G	00	01	FB	003B9	CALLS	#1, NCP\$FAOL	:
00000000'	00	00	90	003C0	MOVB	ENTDC, ENTBUF	1551
		39	11	003CB	BRB	34\$	1544
		BC	D0	003CD	MOVL	@BFR, RO	1565
		60	DD	003D1	PUSHL	(RO)	:
		00	9F	003D3	PUSHAB	ENTDC	:
		AE	9F	003D9	PUSHAB	OUTLEN	:
FFFFFFFF0	8F	5A	D1	003DC	CMPL	CODE, #-16	:
		09	12	003E3	BNEQ	32\$	:
		00	9E	003E5	MOVAB	P.AAY, RO	:
		07	11	003EC	BRB	33\$	:
		00	9E	003EE	MOVAB	P.ABA, RO	32\$:
		50	DD	003F5	PUSHL	RO	33\$:
00000000G	00	04	FB	003F7	CALLS	#4, SYSS\$FAO	:
00000000'	00	AE	90	003FE	MOVB	OUTLEN, ENTBUF	1566
		50	D4	00406	CLRL	RO	1571
		20	BE	95	TSTB	@DTL	:

		0A 13 0040B	BEQL	35\$		
		50 D6 0040D	INCL	R0		
08	AE 00000000'	00 9E 0040F	MOVAB	P.ABC, COMMA		1573
	02	5A D1 00417	35\$:	CMPL	CODE, #2	1578
	01	18 13 0041A	BEQL	36\$		
		13 13 0041C	CMPL	CODE, #1		1580
FFFFFF80	8F	5A D1 0041F	BEQL	36\$		
		5A D1 00421	CMPL	CODE, #-128		1582
	03	0A 13 00428	BEQL	36\$		
		5A D1 0042A	CMPL	CODE, #3		1584
		05 13 0042D	BEQL	36\$		
	14	BE 95 0042F	TSTB	@RSP		1587
		08 12 00432	BNEQ	37\$		
	05	50 E8 00434	36\$:	BLBS	R0, 37\$	1590
	00	BE 95 00437	TSTB	@ERR		1592
		1D 13 0043A	BEQL	38\$		
		6E DD 0043C	37\$:	PUSHL	ERR	1594
	08	AE DD 0043E	PUSHL	ENT		
	28	AE DD 00441	PUSHL	DTL		
	14	AE DD 00444	PUSHL	COMMA		
	24	AE DD 00447	PUSHL	RSP		
		05 DD 0044A	PUSHL	#5		
		8F DD 0044C	PUSHL	#NCP\$ NMLRSP		
00000000G	00	07 FB 00452	CALLS	#7, LIB\$SIGNAL		
	50	5A D0 00459	38\$:	MOVL	CODE, R0	1597
		04 0045C	RET			1599

; Routine Size: 1117 bytes, Routine Base: \$CODE\$ + 0580

```
1614 1600 1 ZSBTTL 'NCP$CONERR Decode an NML Response'
1615 1601 1 GLOBAL ROUTINE NCP$CONERR (COUNT, MSGBFR) = !
1616 1602 1
1617 1603 1 --
1618 1604 1 FUNCTIONAL DESCRIPTION:
1619 1605 1
1620 1606 1 This routine is for the CONNECT routine to have an easy way
1621 1607 1 to process NICE error messages.
1622 1608 1 If the message is an error response, the error is signaled and
1623 1609 1 control does not return to the caller.
1624 1610 1 If the error status is SUC, DON or MOR, and there is a detail
1625 1611 1 or error message, an error is signaled to print these but
1626 1612 1 control returns normally to the caller.
1627 1613 1
1628 1614 1 If an error contains data, it is assumed to be an entity for the
1629 1615 1 error and the entity code is formatted and included in the error
1630 1616 1 message. Entity codes may also occur with success codes and in
1631 1617 1 this case the data is printed as an entity if the message is not
1632 1618 1 a show or list command, indicated by the SHO parameter.
1633 1619 1
1634 1620 1 FORMAL PARAMETERS:
1635 1621 1
1636 1622 1 COUNT Length of buffer containing NICE message
1637 1623 1 MSGBFR Address of buffer containing NICE message
1638 1624 1
1639 1625 1 IMPLICIT INPUTS:
1640 1626 1
1641 1627 1 NCP$GL_ENTITY Entity number sent in original message
1642 1628 1 (If negative, then system-specific entity)
1643 1629 1
1644 1630 1 IMPLICIT OUTPUTS:
1645 1631 1
1646 1632 1 NONE
1647 1633 1
1648 1634 1 ROUTINE VALUE:
1649 1635 1 COMPLETION CODES:
1650 1636 1
1651 1637 1 Value of first byte of message, or error signalled
1652 1638 1
1653 1639 1 SIDE EFFECTS:
1654 1640 1
1655 1641 1 NONE
1656 1642 1
1657 1643 1 --
1658 1644 1
1659 1645 2 BEGIN
1660 1646 2
1661 1647 2 LITERAL
1662 1648 2 RSPSIZ = 32, ! Size of response buffer required
1663 1649 2 DTLSIZ = 32, ! Size of detail buffer required
1664 1650 2 ENTSIZ = 32 ! Size of entity code buffer
1665 1651 2 ;
1666 1652 2
1667 1653 2 LOCAL
1668 1654 2 STATUS, ! Service status return
1669 1655 2 OUTLEN, ! Length in a buffer
1670 1656 2
```

```
: 1671      1657 2      IOSB : BBLOCK [8],      : QIO status
: 1672      1658 2      CTR,                  : General temps
: 1673      1659 2      PTR,
: 1674      1660 2      LEN,
: 1675      1661 2      BFR,
: 1676      1662 2      CODE
: 1677      1663 2      ENTITY,                  : Entity number (negative if sys-specific)
: 1678      1664 2      RSP,                  : Pointer for response text
: 1679      1665 2      SHO,
: 1680      1666 2      COMMA,                  : Pointer to separator before detail
: 1681      1667 2      DTL,                  : Pointer for detail text
: 1682      1668 2      ERR,                  : Pointer for error text
: 1683      1669 2      ENT,                  : Pointer for entity code text
: 1684      1670 2      IDX,                  : Index into tables
: 1685      1671 2      JUNK,                  : Throw away temporary
: 1686      1672 2      DETAIL,                : Value of detail word
: 1687      1673 2      DTLTBL                : Address of detail table
: 1688      1674 2      :
: 1689      1675 2
: 1690      1676 2      OWN
: 1691      1677 2      DTLBUF : VECTOR [DTLSIZ, BYTE], : Detail buffer
: 1692      1678 2      R PBUF : VECTOR [RSPSIZ, BYTE], : Response buffer
: 1693      1679 2      ENTDESC : VECTOR [2],          : Descriptor for string
: 1694      1680 2      ENTBUF : VECTOR [ENTSIZ, BYTE] : Entity string buffer
: 1695      1681 2      :
: 1696      1682 2
: 1697      1683 2      EXTERNAL LITERAL
: 1698      1684 2      NCP$_NMLRSP,            : NML response message
: 1699      1685 2      NCP$_NETIO              : Network communication error
: 1700      1686 2      :
: 1701      1687 2
: 1702      1688 2      EXTERNAL
: 1703      1689 2      NCP$GA_TBL_NMLSTS,        : NML status return codes
: 1704      1690 2      NCP$GA_TBL_FOPDTL,        : File operations detail codes
: 1705      1691 2      NCP$GA_TBL_NCEDTL,        : Network communications detail codes
: 1706      1692 2      NCP$GA_TBL_VMSENTDTL,     : Detail table of VMS specific entities
: 1707      1693 2      NCP$GA_TBL_ENTDTL,        : Detail table of entities
: 1708      1694 2      NCP$GA_TBL_OPEDTL;        : Detail table of operation failures
: 1709      1695 2
: 1710      1696 2
: 1711      1697 2      EXTERNAL ROUTINE
: 1712      1698 2      NCP$FAOSET : NOVALUE,        : Setup to convert entity
: 1713      1699 2      NCP$SHOENTITY : NOVALUE,     : Convert entity
: 1714      1700 2      NCP$FAOL : NOVALUE         : Convert fao string for entity
: 1715      1701 2      :
: 1716      1702 2
: 1717      1703 2
: 1718      1704 2      LEN = 0;                  : Set callers data
: 1719      1705 2      BFR = NCP$GT_RSPBFR;
: 1720      1706 2
: 1721      1707 2      CH$MOVE (.COUNT, .MSGBFR, NCP$GT_RSPBFR); : Copy data into buffer
: 1722      1708 2
: 1723      1709 2
: 1724      1710 2      SHO = 0;
: 1725      1711 2      CTR = .COUNT;            : Point and count into message
: 1726      1712 2      PTR = NCP$GT_RSPBFR;
: 1727      1713 2      :
```



```
: 1728      1714 2 !      We need to set some defaults in case the message is bad
: 1729      1715 2 !
: 1730      1716 2 !
: 1731      1717 2      RSP = UPLIT (%ASCIC 'unrecognized'); ! Some default text for message
: 1732      1718 2      COMMA = UPLIT (%ASCIC '');
: 1733      1719 2      DTL = UPLIT (%ASCIC '');
: 1734      1720 2      ENT = UPLIT (%ASCIC '');
: 1735      1721 2      ERR = UPLIT (%ASCIC '');
: 1736      1722 2
: 1737      1723 2      IF .CTR EQL 0                      ! If message is short, signal now
: 1738      1724 2      THEN
: 1739      1725 2          SIGNAL_STOP (NCP$_NMLRSP, 5, .RSP, .COMMA, .DTL, .ENT, .ERR)
: 1740      1726 2      ;
: 1741      1727 2
: 1742      1728 2      CODE = .(.PTR) <0, 8, 1>;          ! First byte is a code
: 1743      1729 2
: 1744      1730 2      IF NOT NCP$TABLESEARCH              ! Find the code text if possible
: 1745      1731 2      (
: 1746      1732 2          .CODE <0, 8, 0>,                ! Code byte
: 1747      1733 2          NCP$GA_TBL_NMLSTS,              ! Table
: 1748      1734 2          RSP                             ! Return address of counted string
: 1749      1735 2      )
: 1750      1736 2
: 1751      1737 2      THEN
: 1752      1738 2          BEGIN
: 1753      1739 2          $FAO                             ! If not found, make some text
: 1754      1740 2          (
: 1755      1741 2              ASCID ('management return # !SB'),
: 1756      1742 2              OUTLEN,
: 1757      1743 2              UPLIT (RSPSI2-1, RSPBUF+1),
: 1758      1744 2              .CODE
: 1759      1745 2          );
: 1760      1746 2          RSPBUF [0] = .OUTLEN;             ! As a counted string
: 1761      1747 2          RSP = RSPBUF                     ! Point to it
: 1762      1748 2          END
: 1763      1749 2      ;
: 1764      1750 2
: 1765      1751 2      DETAIL = -1;                         ! No detail yet
: 1766      1752 2
: 1767      1753 2      IF .CTR GEQ 3                       ! Is there a detail word
: 1768      1754 2      THEN
: 1769      1755 2          BEGIN
: 1770      1756 2          DETAIL = .(.PTR+1) <0, 16, 1>; ! Obtain the word
: 1771      1757 2          IF .DETAIL NEQ -1                ! Ignore value?
: 1772      1758 2          THEN
: 1773      1759 2              BEGIN
: 1774      1760 2              DTLTBL =                     ! Nope
: 1775      1761 2              BEGIN                          ! Find a table to use
: 1776      1762 2              SELECTONE .CODE OF
: 1777      1763 2              SET
: 1778      1764 2              [NMASC_STS_FOP, NMASC_STS_FIO, NMASC_STS_FCO] :
: 1779      1765 2                  NCP$GA_TBL_FOPDTL          ! File io errors
: 1780      1766 2              ;
: 1781      1767 2              [NMASC_STS_MLD, NMASC_STS_MCF] :
: 1782      1768 2                  NCP$GA_TBL_NCEDTL          ! Network io errors
: 1783      1769 2              ;
: 1784      1770 2              [NMASC_STS_OPE] :
```

```
1785 1771 5      NCP$GA_TBL_OPEDTL      ! Operation failure
1786 1772 5
1787 1773 5      [NMA$C_STS_CMP, NMA$C_STS_IDE, NMA$C_STS_STA] :
1788 1774 5      ! Errors with entities
1789 1775 5      IF .NCP$GL_ENTITY LSS 0 ! If system-specific entity
1790 1776 5      THEN
1791 1777 5          NCP$GA_TBL_VMSENTDTL      ! VMS entities
1792 1778 5      ELSE
1793 1779 5          NCP$GA_TBL_ENTDTL;          ! DNA entities
1794 1780 5      [OTHERWISE] :          ! Details not valid
1795 1781 6      BEGIN
1796 1782 6      IF .DETAIL EQL 0          ! Zero is null detail here
1797 1783 6      THEN 1                  ! Null detail if not valid
1798 1784 6      ELSE 0                  ! But report non zero detail
1799 1785 6      END
1800 1786 5      ;
1801 1787 5      YES
1802 1788 5      END
1803 1789 4      ;
1804 1790 4
1805 1791 4      IF .CODE EQL NMA$C_STS_OPE      ! If operation failure
1806 1792 4      AND
1807 1793 5      (.NCP$GL_ENTITY EQL          ! and entity is line
1808 1794 5      NMA$C_ENT_LIN
1809 1795 5      OR
1810 1796 5      .NCP$GL_ENTITY EQL          ! or circuit
1811 1797 5      NMA$C_ENT_CIR)
1812 1798 5
1813 1799 4      THEN
1814 1800 5      BEGIN
1815 1801 5      LOCAL
1816 1802 5      PREBUF : VECTOR [40, BYTE],    ! Buffer for string to proceed
1817 1803 5      ! each detail message.
1818 1804 5      PRELEN,                          ! Length of string to proceed
1819 1805 5      ! each detail message.
1820 1806 5      LOCPTR;                          ! Local pointer
1821 1807 5
1822 1808 5      LOCPTR = PREBUF;                ! Init pointer into buffer
1823 1809 5
1824 1810 5      ;
1825 1811 5      Build the string which will precede the detail text so that each detail
1826 1812 5      string output will line-up under the error text.  For example:
1827 1813 5
1828 1814 5      %facility-L-ident, error text      ! Original error message
1829 1815 5
1830 1816 5      %facility-L-ident, error text<CR><LF> ! Message with two detail
1831 1817 5      < SPACES >, detail text<CR><LF>    ! strings appended.
1832 1818 5      < SPACES >, detail text
1833 1819 5
1834 1820 6      PRELEN = ( CH$FIND CH( (.PTR+3), ! Get the number of characters
1835 1821 6      .PTR + 4, %C' ') ) ! in the facility and ident
1836 1822 5      - (.PTR + 4); ! portion of error message
1837 1823 5
1838 1824 5      .LOCPTR <0, 16> = %X'0A0D'; ! Store <CR><LF> in buffer,
1839 1825 5      LOCPTR = CH$FILL( %C' ', .PRELEN, .LOCPTR + 2 ); ! some spaces,
1840 1826 5      .LOCPTR <0, 16> = %ASCII', ' '; ! and a ' '
1841 1827 5      PRELEN = .PRELEN + 4; ! Length = length of facility
```



```
: 1842      1828 5                                ! text plus <CR><LF> and "", ""
: 1843      1829 5
: 1844      1830 5      LOCPTR = .PTR + 4 +      ! Point to end of original
: 1845      1831 5      (.PTR + 3) < 0, 8 >; ! error message text.
: 1846      1832 5
: 1847      1833 5      INCR INDEX FROM 0 TO 16 DO
: 1848      1834 6      BEGIN
: 1849      1835 6      IF .DETAIL < .INDEX, 1, 0 > ! If status or error bit is set,
: 1850      1836 6      AND                        ! and it's in the table,
: 1851      1837 6      NCP$TABLESEARCH (.INDEX, .DTLTBL, DTL)
: 1852      1838 6      AND                        ! and there's room in the
: 1853      1839 6      .PRELEN + (.DTL) < 0, 8 > ! response buffer.
: 1854      1840 6      LEQ .PTR + NCP$C_RSPSI2 - .LOCPTR
: 1855      1841 6      THEN
: 1856      1842 7      BEGIN
: 1857      1843 7      LOCPTR = CH$MOVE      ! Append the string which
: 1858      1844 7      (                      ! preceeds each detail message
: 1859      1845 7      .PRELEN,              ! to the end of the error
: 1860      1846 7      PREBUF,               ! message
: 1861      1847 7      .LOCPTR
: 1862      1848 7      );
: 1863      1849 7
: 1864      1850 7      LOCPTR = CH$MOVE      ! Append detail to end of the
: 1865      1851 7      (                      ! error message
: 1866      1852 7      (.DTL) < 0, 8 >,
: 1867      1853 7      .DTL + 1,
: 1868      1854 7      .LOCPTR
: 1869      1855 7      );
: 1870      1856 7
: 1871      1857 6      END;
: 1872      1858 5      END;
: 1873      1859 5
: 1874      1860 5      (.PTR + 3) < 0, 8 > =      ! Update message length.
: 1875      1861 5      .LOCPTR - .PTR - 4;
: 1876      1862 5      CTR < 0, 8 > = .LOCPTR - .PTR; ! Update counter.
: 1877      1863 5      DTLTBL = 1;              ! Indicate that we formatted it
: 1878      1864 5      DTLBUF [0] = 0;          ! Make sure we Don't print the
: 1879      1865 5      DTL = DTLBUF;            ! detail #
: 1880      1866 5
: 1881      1867 5      END
: 1882      1868 5
: 1883      1869 4      ELSE
: 1884      1870 4      IF .CODE EQL NMA$C_STS_PVA      ! Special details for these
: 1885      1871 4      OR                          ! Errors, its the parameter
: 1886      1872 4      .CODE EQL NMA$C_STS_PLO      ! name
: 1887      1873 4      OR
: 1888      1874 4      .CODE EQL NMA$C_STS_PNA
: 1889      1875 4      OR
: 1890      1876 4      .CODE EQL NMA$C_STS_PTY
: 1891      1877 4      OR
: 1892      1878 4      .CODE EQL NMA$C_STS_PGP
: 1893      1879 4      OR
: 1894      1880 4      .CODE EQL NMA$C_STS_PMS
: 1895      1881 4      THEN
: 1896      1882 5      BEGIN
: 1897      1883 5      NCP$FORMATPARM      ! Format the parameter name
: 1898      1884 5      (
```

```

: 1899      1885 5      .NCP$GL_ENTITY,      ! Entity is here
: 1900      1886 5      DETAIL,              ! Parameter code is here
: 1901      1887 5      TRUE,                ! Give the name
: 1902      1888 5      FALSE,              ! Not the data
: 1903      1889 5      UPLIT (DTLSIZ - 1, DTLBUF + 1), ! Describe the buffer
: 1904      1890 5      OUTLEN,              ! Length of text here
: 1905      1891 5      JUNK,                ! Return pointer to throw away
: 1906      1892 5      );
: 1907      1893 5      DTLBUF [0] = .OUTLEN; ! Set length of counted string
: 1908      1894 5      DTL = DTLBUF;         ! Point to buffer
: 1909      1895 5      DTLTBL = 1           ! Kill following check
: 1910      1896 5      END
: 1911      1897 4      ;
: 1912      1898 4      ;
: 1913      1899 4      IF .DTLTBL NEQ 1      ! Unless we formatted it above
: 1914      1900 4      AND
: 1915      1901 5      (
: 1916      1902 5      .DTLTBL EQL 0         ! If there is no detail table
: 1917      1903 5      OR
: 1918      1904 6      (
: 1919      1905 6      IF .DTLTBL NEQ 0      ! Interlock for not in table check
: 1920      1906 6      THEN
: 1921      1907 6      NOT NCP$TABLESEARCH (.DETAIL, .DTLTBL, DTL)
: 1922      1908 6      ELSE
: 1923      1909 6      TRUE                  ! Force conversion if not in table
: 1924      1910 6      )
: 1925      1911 5      )
: 1926      1912 4      THEN
: 1927      1913 5      BEGIN                  ! Put out in some standard way
: 1928      1914 5      $FAO
: 1929      1915 5      (
: 1930      1916 5      ASCID ('detail # !UW'),
: 1931      1917 5      OUTLEN,
: 1932      1918 5      UPLIT (DTLSIZ-1, DTLBUF+1),
: 1933      1919 5      .DETAIL
: 1934      1920 5      );
: 1935      1921 5      DTLBUF [0] = .OUTLEN; ! As counted string
: 1936      1922 5      DTL = DTLBUF
: 1937      1923 5      END
: 1938      1924 4      END
: 1939      1925 3      END
: 1940      1926 2      ;
: 1941      1927 2      ;
: 1942      1928 2      IF .CTR GEQU 4        ! If there is enough for system
: 1943      1929 2      THEN                  ! Specific error text
: 1944      1930 3      BEGIN
: 1945      1931 4      IF .CTR GEQU (4 + .(.PTR+3) < 0, 8, 0) )
: 1946      1932 3      THEN                  ! And the text is valid
: 1947      1933 4      BEGIN
: 1948      1934 4      ERR = .PTR + 3;        ! Point to the counted string
: 1949      1935 4      LEN = .CTR - (.(.PTR+3) < 0, 8, 0) - 4; ! Adjust returned length
: 1950      1936 5      BFR = .BFR + 4 + (.(.PTR+3) < 0, 8, 0) ! And buffer beyond it
: 1951      1937 4      END
: 1952      1938 3      ELSE                  ! Tell the world its not clean
: 1953      1939 3      ERR = UPLIT (%ASCIC '%NCP-W-ERRRSP, invalid error text in listener response')
: 1954      1940 3      END
: 1955      1941 2      ;
```

```
1956 1942 2
1957 1943 2
1958 1944 2
1959 1945 2
1960 1946 2
1961 1947 2
1962 1948 2
1963 1949 2
1964 1950 2
1965 1951 2
1966 1952 3
1967 1953 3
1968 1954 3
1969 1955 3
1970 1956 3
1971 1957 3
1972 1958 3
1973 1959 4
1974 1960 4
1975 1961 4
1976 1962 4
1977 1963 4
1978 1964 4
1979 1965 4
1980 1966 3
1981 1967 4
1982 P 1968 4
1983 P 1969 4
1984 P 1970 4
1985 P 1971 4
1986 P 1972 4
1987 P 1973 4
1988 P 1974 4
1989 P 1975 4
1990 P 1976 4
1991 P 1977 4
1992 1978 4
1993 1979 4
1994 1980 4
1995 1981 3
1996 1982 2
1997 1983 2
1998 1984 2
1999 1985 2
2000 1986 2
2001 1987 2
2002 1988 2
2003 1989 3
2004 1990 4
2005 1991 4
2006 1992 4
2007 1993 4
2008 1994 4
2009 1995 4
2010 1996 4
2011 1997 4
2012 1998 4

Signal the error to print it

IF .LEN NEQ 0
AND
NOT .SHO
THEN
BEGIN
LEN = 0;
ENTDSC [0] = ENTSIZ - 1;
ENTDSC [1] = ENTBUF + 1;
ENT = ENTBUF;
IF .NCP$GL_FNC_CODE NEQ NMASC_FNC_TES ! Loop return with test data
THEN
BEGIN
PTR = .BFR;
NCP$FAOSET ();
NCP$SHOENTITY (PTR);
NCP$FAOL (ENTDSC);
ENTBUF [0] = .ENTDSC [0];
END
ELSE
BEGIN
$FAO
(
IF .CODE EQL NMASC_STS_PVA ! Special case the text for
THEN ASCID ('Maximum data length = !UW') ! a loop message
ELSE ASCID ('Messages not looped = !UW')
),
OUTLEN,
ENTDSC,
.BFR
);
ENTBUF [0] = .OUTLEN
END
END

;
IF CH$RCHAR(.DTL) NEQ 0
THEN
COMMA = UPLIT(%ASCIC ',');

IF
(
.CODE NEQ NMASC_STS_MOR
AND
.CODE NEQ NMASC_STS_SUC
AND
.CODE NEQ NMASC_STS_DON
AND
.CODE NEQ NMASC_STS_PAR
)
! Is there an entity for the message
! and this is not a show or list
! Return no data to caller
! Descriptor for output is buffer
! Less one byte for count
! Set counted string address
! Loop return with test data
! Set pointer to entity code
! Setup conversion routines
! Convert to fao parameters
! Convert to text
! Make counted string
! Convert test data if loop return
! Descriptor of buffer
! Stack the data (word of loop count)
! Set counter for this message
! If text following message,
! then delimit with a comma
! If a not a success code
```

```
: 2013      1999      3      AND
: 2014      2000      3      CH$RCHAR (.RSP) NEQ 0      ! and the response message is here
: 2015      2001      3      )
: 2016      2002      2      OR
: 2017      2003      2      CH$RCHAR (.DTL) NEQ 0      ! or any of the text strings are here
: 2018      2004      2      OR
: 2019      2005      2      CH$RCHAR (.ERR) NEQ 0      ! then print the error
: 2020      2006      2      THEN
: 2021      2007      2      SIGNAL (NCP$_NMLRSP, 5, .RSP, .COMMA, .DTL, .ENT, .ERR)
: 2022      2008      2      ;
: 2023      2009      2
: 2024      2010      2      RETURN .CODE      ! Return data to caller
: 2025      2011      2
: 2026      2012      1      END;
```

```
.PSECT $SPLITS,NOWRT,NOEXE,2
00 00 64 65 7A 69 6E 67 6F 63 65 72 6E 75 0C 0014C P.ABD: .ASCII <12>\unrecognized\<0><0><0>
                                00 0015B
                                00 00 00 00 0015C P.ABE: .ASCII <0><0><0><0>
                                00 00 00 00 00160 P.ABF: .ASCII <0><0><0><0>
                                00 00 00 00 00164 P.ABG: .ASCII <0><0><0><0>
                                00 00 00 00 00168 P.ABH: .ASCII <0><0><0><0>
75 74 65 72 20 74 6E 65 6D 65 67 61 6E 61 6D 0016C P.ABJ: .ASCII \management return # !SB\<0>
                                00 42 53 21 20 23 20 6E 72 0017B
                                00000017 00184 P.ABI: .LONG 23
                                00000000 00188 .ADDRESS P.ABJ
                                0000001F 0018C P.ABK: .LONG 31
                                00000000 00190 .ADDRESS RSPBUF+1
                                0000001F 00194 P.ABL: .LONG 31
                                00000000 00198 .ADDRESS DTLBUF+1
                                57 55 21 20 23 20 6C 69 61 74 65 64 0019C P.ABN: .ASCII \detail # !UW\
                                0000000C 001A8 P.ABM: .LONG 12
                                00000000 001AC .ADDRESS P.ABN
                                0000001F 001B0 P.ABO: .LONG 31
                                00000000 001B4 .ADDRESS DTLBUF+1
2C 50 53 52 52 52 45 2D 57 2D 50 43 4E 25 36 001B8 P.ABP: .ASCII \6XNCP-W-ERRRSP, invalid error text in li\
20 72 6F 72 72 65 20 64 69 6C 61 76 6E 69 20 001C7
65 73 6E 6F 70 73 65 72 2D 72 65 6E 65 74 73 001D6
                                00 001E0 .ASCII \stener response\<0>
                                00 001EF
65 6C 20 61 74 61 64 20 6D 75 6D 69 78 61 4D 001F0 P.ABR: .ASCII \Maximum data length = !UW\<0><0><0>
                                00 00 00 57 55 21 20 3D 20 68 74 67 6E 001FF
                                00000019 0020C P.ABQ: .LONG 25
                                00000000 00210 .ADDRESS P.ABR
6F 6C 20 74 6F 6E 20 73 65 67 61 73 73 65 4D 00214 P.ABT: .ASCII \Messages not looped = !UW\<0><0><0>
                                00 00 00 57 55 21 2C 3D 20 64 65 70 6F 00223
                                00000019 00230 P.ABS: .LONG 25
                                00000000 00234 .ADDRESS P.ABT
                                00 00 2C 01 00238 P.ABU: .ASCII <1>\,\<0><0>

.PSECT $OWNS,NOEXE,2
000E4 DTLBUF: .BLKB 32
00104 RSPBUF: .BLKB 32
```

00124 ENTDSO: .BLKB 8
0012C ENTBUF: .BLKB 32

				.PSECT	\$CODE\$,NOWRT,2	
				.ENTRY	NCP\$CONERR, Save R2,R3,R4,R5,R6,R7,R8,R9,-	
				OFFC	00000	
	5E	A0	AE 9E 00002	MOVAB	R10,R11	1601
			58 D4 00006	CLRL	-96(SP), SP	
	5B	00000000'	00 9E 00008	MOVAB	LEN	1704
00000000' 00	08	BC 04	AC 28 0000F	MOVAB	NCP\$GT_RSPBFR, BFR	1705
		0C	AE D4 00019	MOVAB	COUNT, -@MSGBFR, NCP\$GT_RSPBFR	1707
	57	04	AC D0 0001C	CLRL	SHO	1709
	28	AE 00000000'	00 9E 00020	MOVL	COUNT, CTR	1710
	18	AE 00000000'	00 9E 00028	MOVAB	NCP\$GT_RSPBFR, PTR	1711
	08	AE 00000000'	00 9E 00030	MOVAB	P.ABD, RSP	1717
	24	AE 00000000'	00 9E 00038	MOVAB	P.ABE, COMMA	1718
	04	AE 00000000'	00 9E 00040	MOVAB	P.ABF, DTL	1719
		6E 00000000'	00 9E 00048	MOVAB	P.ABG, ENT	1720
			57 D5 0004F	MOVAB	P.ABH, ERR	1721
			1D 12 00051	TSTL	CTR	1723
			6E DD 00053	BNEQ	1\$	
	08		AE DD 00055	PUSHL	ERR	1725
	2C		AE DD 00058	PUSHL	ENT	
	14		AE DD 0005B	PUSHL	DTL	
	28		AE DD 0005E	PUSHL	COMMA	
			05 DD 00061	PUSHL	RSP	
		00000000G	8F DD 00063	PUSHL	#5	
00000000G	00		07 FB 00069	PUSHL	#NCP\$ NMLRSP	
	59	28	AE D0 00070	CALLS	#7, LIB\$STOP	
10	AE		69 98 00074	MOVL	PTR, R9	1728
		18	AE 9F 00078	CVTBL	(R9), CODE	
		00000000G	00 9F 0007B	PUSHAB	RSP	1731
	7E	18	AE 9A 00081	PUSHAB	NCP\$GA TBL NMLSTS	
00000000V	00		03 FB 00085	MOVZBL	CODE, =(SP)	1732
	29		50 E8 0008C	CALLS	#3, NCP\$TABLESEARCH	
		10	AE DD 0008F	BLBS	R0, 2\$	
		00000000'	00 9F 00092	PUSHL	CODE	1745
		34	AE 9F 00098	PUSHAB	P.ABK	
		00000000'	00 9F 0009B	PUSHAB	OUTLEN	
			04 FB 000A1	PUSHAB	P.ABI	
00000000G	00		01 CE 000B8	CALLS	#4, SYSS\$FAO	
00000000'	00	2C	AE 90 000A8	MOVAB	OUTLEN, RSPBUF	1746
	18		00 9E 000B0	MOVAB	RSPBUF, RSP	1747
	20		01 CE 000B8	MNEGL	#1, DETAIL	1751
		03	57 D1 000BC	CMPL	CTR, #3	1753
			03 18 000BF	BGEQ	4\$	
		0217	31 000C1	BRW	24\$	
	20	AE 01	A9 32 000C4	CVTWL	1(R9), DETAIL	1756
FFFFFFFFF	8F	20	AE D1 000C9	CMPL	DETAIL, #-1	1757
			EE 13 000D1	BEQL	3\$	
FFFFFFFFE	8F	10	AE D1 000D3	CMPL	CODE, #-18	1764
			14 13 000DB	BEQL	5\$	
FFFFFFFF2	8F	10	AE D1 000DD	CMPL	CODE, #-14	
			13 19 000E5	BLSS	6\$	
FFFFFFFF3	8F	10	AE D1 000E7	CMPL	CODE, #-13	

			09	14	000EF		BGTR	6\$		
		5A	00	9E	000F1	5\$:	MOVAB	NCP\$GA_TBL_FOPDTL, DTLTBL		
			74	11	000F8		BRB	14\$		
FFFFFFEB	8F	10	AE	D1	000FA	6\$:	CMPL	CODE, #-21	1767	
			0A	13	00102		BEQL	7\$		
FFFFFFED	8F	10	AE	D1	00104		CMPL	CODE, #-19		
			09	12	0010C		BNEQ	8\$		
		5A	00	9E	0010E	7\$:	MOVAB	NCP\$GA_TBL_NCEDTL, DTLTBL		
			57	11	00115		BRB	14\$		
FFFFFFE7	8F	10	AE	D1	00117	8\$:	CMPL	CODE, #-25	1770	
			09	12	0011F		BNEQ	9\$		
		5A	00	9E	00121		MOVAB	NCP\$GA_TBL_OPEDTL, DTLTBL		
			44	11	00128		BRB	14\$		
FFFFFFF5	8F	10	AE	D1	0012A	9\$:	CMPL	CODE, #-11	1773	
			14	13	00132		BEQL	10\$		
FFFFFFF7	8F	10	AE	D1	00134		CMPL	CODE, #-9		
			24	19	0013C		BLSS	12\$		
FFFFFFF8	8F	10	AE	D1	0013E		CMPL	CODE, #-8		
			1A	14	00146		BGTR	12\$		
			00	D5	00148	10\$:	TSTL	NCP\$GL_ENTITY	1775	
			09	18	0014E		BGEQ	11\$		
		5A	00	9E	00150		MOVAB	NCP\$GA_TBL_VMSENTDTL, DTLTBL		
			15	11	00157		BRB	14\$		
		5A	00	9E	00159	11\$:	MOVAB	NCP\$GA_TBL_ENTDTL, DTLTBL		
			0C	11	00160		BRB	14\$		
		20	AE	D5	00162	12\$:	TSTL	DETAIL	1782	
			05	12	00165		BNEQ	13\$		
		5A	01	D0	00167		MOVL	#1, DTLTBL		
			02	11	0016A		BRB	14\$		
			5A	D4	0016C	13\$:	CLRL	DTLTBL		
FFFFFFE7	8F	10	AE	D1	0016E	14\$:	CMPL	CODE, #-25	1791	
			0F	12	00176		BNEQ	15\$		
		50	00	D0	00178		MOVL	NCP\$GL_ENTITY, R0	1793	
		01	50	D1	0017F		CMPL	R0, #1		
			08	13	00182		BEQL	16\$		
		03	50	D1	00184		CMPL	R0, #3	1796	
			03	13	00187	15\$:	BEQL	16\$		
			009D	31	00189		BRW	20\$		
		53	AE	9E	0018C	16\$:	MOVAB	PREBUF, LOCPTR	1808	
04	A9	03	2C	3A	00190		LOCC	#44, 3(R9), 4(R9)	1820	
			02	12	00196		BNEQ	17\$		
			51	D4	00198		CLRL	R1		
		50	A9	9E	0019A	17\$:	MOVAB	4(R9), R0	1822	
		56	50	C3	0019E		SUBL3	R0, R1, PRELEN		
			50	3C	001A2		MOVZWL	LOCPTR, R0	1824	
		60	8F	3C	001A5		MOVZWL	#2573, (R0)		
56	20		0A0D	00	2C	001AA	MOVCS	#0, (SP), #32, PRELEN, 2(LOCPTR)	1825	
			02	A3	001AF					
		50	53	3C	001B1		MOVZWL	LOCPTR, R0	1826	
		60	8F	3C	001B4		MOVZWL	#8236, (R0)		
		56	04	C0	001B9		ADDL2	#4, PRELEN	1827	
		50	A9	9A	001BC		MOVZBL	3(R9), R0	1831	
		53	04	A049	9E	001C0	MOVAB	4(R0)[R9], LOCPTR	1830	
			14	AE	D4	001C5	CLRL	INDEX	1833	
	38	20	AE	E1	001C8	18\$:	BBC	INDEX, DETAIL, 19\$	1835	
			24	AE	9F	001CE	PUSHAB	DTL	1837	
			5A	DD	001D1		PUSHL	DTLTBL		

			1C	AE	DD	001D3	PUSHL	INDEX		
	00000000V	00		03	FB	001D6	CALLS	#3, NCP\$TABLESEARCH		
		26		50	E9	001DD	BLBC	R0, 19\$		
		51	24	BE	9A	001E0	MOVZBL	DTL, R1		1839
		51		56	C0	001E4	ADDL2	PRELEN, R1		
50		59		53	C3	001E7	SUBL3	LOCPTR, R9, R0		1840
		50	03E8	C0	9E	001EB	MOVAB	1000(R0), R0		
		50		51	D1	001F0	CMPL	R1, R0		
				11	14	001F3	BGTR	19\$		
63	30	AE		56	28	001F5	MOV3	PRELEN, PREBUF, (LOCPTR)		1847
		50	24	AE	D0	001FA	MOVL	DTL, R0		1853
		51		60	9A	001FE	MOVZBL	(R0), R1		
63	01	A0		51	28	00201	MOV3	R1, 1(R0), (LOCPTR)		1855
BD	14	AE		10	F3	00206	A0BLEQ	#16, INDEX, 18\$		1833
		53		59	C2	0020B	SUBL2	R9, R3		1861
03	A9	53		04	83	0020E	SUBB3	#4, R3, 3(R9)		
		57		53	90	00213	MOVB	R3, CTR		1862
		5A		01	D0	00216	MOVL	#1, DTLTBL		1863
			00000000'	00	94	00219	CLRB	DTLBUF		1864
	24	AE	00000000'	00	9E	0021F	MOVAB	DTLBUF, DTL		1865
				6E	11	00227	BRB	22\$		1791
FFFFFFF0	8F		10	AE	D1	00229	CMPL	CODE, #-16		1870
				32	13	00231	BEQL	21\$		
FFFFFFE9	8F		10	AE	C1	00233	CMPL	CODE, #-23		1872
				28	13	0023B	BEQL	21\$		
FFFFFFEA	8F		10	AE	D1	0023D	CMPL	CODE, #-22		1874
				1E	13	00245	BEQL	21\$		
FFFFFFFA	8F		10	AE	D1	00247	CMPL	CODE, #-6		1876
				14	13	0024F	BEQL	21\$		
FFFFFFE5	8F		10	AE	D1	00251	CMPL	CODE, #-27		1878
				0A	13	00259	BEQL	21\$		
FFFFFFE3	8F		10	AE	D1	0025B	CMPL	CODE, #-29		1880
				32	12	00263	BNEQ	22\$		
		1C		AE	9F	00265	PUSHAB	JUNK		1884
		30		AE	9F	00268	PUSHAB	OUTLEN		
			00000000'	00	9F	0026B	PUSHAB	P.ABL		1889
		7E		01	7D	00271	MOVQ	#1, -(SP)		1884
			34	AE	9F	00274	PUSHAB	DETAIL		
			00000000G	00	DD	00277	PUSHL	NCP\$GL ENTITY		1885
00000000G	00			07	FB	0027D	CALLS	#7, NCP\$FORMATPARM		
00000000'	00		2C	AE	90	00284	MOVB	OUTLEN, DTLBUF		1893
	24	AE	00000000'	00	9E	0028C	MOVAB	DTLBUF, DTL		1894
		5A		01	D0	00294	MOVL	#1, DTLTBL		1895
		01		5A	D1	00297	CMPL	DTLTBL, #1		1899
				3F	13	0029A	BEQL	24\$		
				5A	D5	0029C	TSTL	DTLTBL		1902
				12	13	0029E	BEQL	23\$		
		24		AE	9F	002A0	PUSHAB	DTL		1907
				5A	DD	002A3	PUSHL	DTLTBL		
		28		AE	DD	002A5	PUSHL	DETAIL		
00000000V	00			03	FB	002A8	CALLS	#3, NCP\$TABLESEARCH		
	29			50	E8	002AF	BLBS	R0, 24\$		
		20		AE	DD	002B2	PUSHL	DETAIL		1920
			00000000'	00	9F	002B5	PUSHAB	P.ABO		
		34		AE	9F	002BB	PUSHAB	OUTLEN		
			00000000'	00	9F	002BE	PUSHAB	P.ABM		
00000000G	00			04	FB	002C4	CALLS	#4, SYSSFAO		

00000000'	00	2C	AE	90	002CB	MOV	OUTLEN, DTLBUF	1921
24	AE	00000000'	00	9E	002D3	MOVAB	DTLBUF, DTL	1922
	04		57	D1	002DB	CMPL	CTR, #4	1928
			26	1F	002DE	BLSSU	26\$	
	50	03	A9	9A	002E0	MOVZBL	3(R9), R0	1931
	51	04	A0	9E	002E4	MOVAB	4(R0), R1	
	51		57	D1	002E8	CMPL	CTR, R1	
			12	1F	002EB	BLSSU	25\$	
	6E	03	A9	9E	002ED	MOVAB	3(R9), ERR	1934
	57		50	C2	002F1	SUBL2	R0, R7	1935
	58	FC	A7	9E	002F4	MOVAB	-4(R7), LEN	
	5B	04	A04B	9E	002F8	MOVAB	4(R0)[BFR], BFR	1936
			07	11	002FD	BRB	26\$	
	6E	00000000'	00	9E	002FF	MOVAB	P.ABP, ERR	1939
			58	D5	00306	TSTL	LEN	1948
			56	13	00308	BEQL	27\$	
	52	0C	AE	E8	0030A	BLBS	SHO, 27\$	1950
			58	D4	0030E	CLRL	LEN	1953
00000000'	00		1F	D0	00310	MOVL	#31, ENTDC	1954
00000000'	00	00000000'	00	9E	00317	MOVAB	ENTBUF+1, ENTDC+4	1955
04	AE	00000000'	00	9E	00322	MOVAB	ENTBUF, ENT	1956
	12	00000000G	00	D1	0032A	CMPL	NCP\$GL_FNC_CODE, #18	1957
			2F	13	00331	BEQL	28\$	
28	AE		5B	D0	00333	MOVL	BFR, PTR	1960
00000000G	00		00	FB	00337	CALLS	#0, NCP\$FAOSET	1961
		28	AE	9F	0033E	PUSHAB	PTR	1962
00000000G	00		01	FB	00341	CALLS	#1, NCP\$SHOENTITY	
		00000000'	00	9F	00348	PUSHAB	ENTDC	1963
00000000G	00		01	FB	0034E	CALLS	#1, NCP\$FAOL	
00000000'	00	00000000'	00	90	00355	MOV	ENTDC, ENTBUF	1964
			36	11	00360	BRB	31\$	1957
			6B	DD	00362	PUSHL	(BFR)	1978
		00000000'	00	9F	00364	PUSHAB	ENTDC	
		34	AE	9F	0036A	PUSHAB	OUTLEN	
FFFFFFFF0	8F	1C	AE	D1	0036D	CMPL	CODE, #-16	
			09	12	00375	BNEQ	29\$	
	50	00000000'	00	9E	00377	MOVAB	P.ABQ, R0	
			07	11	0037E	BRB	30\$	
	50	00000000'	00	9E	00380	MOVAB	P.ABS, R0	
			50	DD	00387	PUSHL	R0	
00000000G	00		04	FB	00389	CALLS	#4, SYSS\$FAO	
00000000'	00	2C	AE	90	00390	MOV	OUTLEN, ENTBUF	1979
			50	D4	00398	CLRL	R0	1984
		24	BE	95	0039A	TSTB	@DTL	
			0A	13	0039D	BEQL	32\$	
			50	D6	0039F	INCL	R0	
08	AE	00000000'	00	9E	003A1	MOVAB	P.ABU, COMMA	1986
	02	10	AE	D1	003A9	CMPL	CODE, #2	1991
			1B	13	003AD	BEQL	33\$	
	01	10	AE	D1	003AF	CMPL	CODE, #1	1993
			15	13	003B3	BEQL	33\$	
FFFFFFFF80	8F	10	AE	D1	003B5	CMPL	CODE, #-128	1995
			0B	13	003BD	BEQL	33\$	
	03	10	AE	D1	003BF	CMPL	CODE, #3	1997
			05	13	003C3	BEQL	33\$	
		18	BE	95	003C5	TSTB	@RSP	2000
			08	12	003C8	BNEQ	34\$	

NCPNETIO
V04-000

Network I/O Routines
NCP\$CONERR Decode an NML Response

H 4
15-Sep-1984 23:46:44
14-Sep-1984 12:48:14

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[NCP.SRC]NCPNETIO.B32;1 (15)

Page 63

NCP
V04

05		50	E8	003CA	33\$:	BLBS	R0, 34\$	:	2003
	00	BE	95	003CD		TSTB	@ERR	:	2005
		1D	13	003D0		BEQL	35\$	:	
		6E	DD	003D2	34\$:	PUSHL	ERR	:	2007
	08	AE	DD	003D4		PUSHL	ENT	:	
	2C	AE	DD	003D7		PUSHL	DTL	:	
	14	AE	DD	003DA		PUSHL	COMMA	:	
	28	AE	DD	003DD		PUSHL	RSP	:	
		05	DD	003E0		PUSHL	#5	:	
00000000G	00	8F	DD	003E2		PUSHL	#NCP\$ NMLRSP	:	
	50	07	FB	003E8		CALLS	#7, LIB\$SIGNAL	:	
		10	AE	DD	003EF	35\$:	MOVL	:	2010
			04	003F3		RET	CODE, R0	:	2012

; Routine Size: 1012 bytes. Routine Base: \$CODE\$ + 09DD

```
: 2028 2013 1 %SBTTL 'NCP$TABLESEARCH Find an Entry in a Text Table'
: 2029 2014 1 GLOBAL ROUTINE NCP$TABLESEARCH (CODE, TBL, RTXTC) =
: 2030 2015 1
: 2031 2016 1
: 2032 2017 1 ++
: 2033 2018 1 FUNCTIONAL DESCRIPTION:
: 2034 2019 1 This routine searches a table for a word code and returns an
: 2035 2020 1 address of a counted string of an associated text string.
: 2036 2021 1
: 2037 2022 1 FORMAL PARAMETERS:
: 2038 2023 1
: 2039 2024 1 CODE Value of the code word
: 2040 2025 1 TBL Address of the table
: 2041 2026 1 RTXTC Address to return the address of the counted string
: 2042 2027 1
: 2043 2028 1 IMPLICIT INPUTS:
: 2044 2029 1
: 2045 2030 1 NONE
: 2046 2031 1
: 2047 2032 1 IMPLICIT OUTPUTS:
: 2048 2033 1
: 2049 2034 1 NONE
: 2050 2035 1
: 2051 2036 1 ROUTINE VALUE:
: 2052 2037 1 COMPLETION CODES:
: 2053 2038 1
: 2054 2039 1 Success or failure RTXTC set to 'unrecognized' if failure
: 2055 2040 1
: 2056 2041 1 SIDE EFFECTS:
: 2057 2042 1
: 2058 2043 1 NONE
: 2059 2044 1
: 2060 2045 1 --
: 2061 2046 1
: 2062 2047 2 BEGIN
: 2063 2048 2
: 2064 2049 2 LOCAL TPTR : REF BBLOCKVECTOR [1, 4] ! Pointer to the table
: 2065 2050 2 TPTR : REF BBLOCKVECTOR [1, 4]
: 2066 2051 2 ;
: 2067 2052 2
: 2068 2053 2 .RTXTC = UPLIT (%ASCIC 'unrecognized');
: 2069 2054 2 TPTR = .TBL;
: 2070 2055 2
: 2071 2056 2 INCRU IDX FROM 0 ! Scan the table
: 2072 2057 2 DO
: 2073 2058 3 BEGIN
: 2074 2059 3 IF .TPTR [.IDX, 0, 0, 16, 1] ! Look for the end first
: 2075 2060 3 EQL ! Use a signed reference for this
: 2076 2061 3 -1
: 2077 2062 3 THEN
: 2078 2063 3 RETURN FAILURE ! Not found, return failure
: 2079 2064 3 ;
: 2080 2065 3
: 2081 2066 3 IF .TPTR [.IDX, 0, 0, 16, 0] ! Look for the code (unsigned)
: 2082 2067 3 EQL
: 2083 2068 3 .CODE <0, 16, 0> ! Code as a word
: 2084 2069 3 THEN
```

```
: 2085      2070 4      BEGIN
: 2086      2071 4      .RTXTC = .TPTR [ .IDX, 2, 0, 16, 1] ! Return the real address
: 2087      2072 4      + TPTR [ .IDX, 2, 0, 16, 1] ! Make address from the offset
: 2088      2073 4      RETURN SUCCESS
: 2089      2074 4      END
: 2090      2075 3      END
: 2091      2076 2      .
: 2092      2077 2      RETURN FAILURE
: 2093      2078 2
: 2094      2079 1      END;
                                ! Better never fail this way
```

```
                                .PSECT $SPLITS$,NOWRT,NOEXE,2
00 00 64 65 7A 69 6E 67 6F 63 65 72 6E 75 0C 0023C P.ABV: .ASCII <12>\unrecognized\<0><0><0>
                                00 0024B
```

```
                                .PSECT $CODE$,NOWRT,2
                                .ENTRY NCP$TABLESEARCH, Save R2,R3
                                MOVAB P.ABV, @RTXTC
                                MOVL TBL, TPTR
                                CLRL IDX
                                MOVAL (TPTR)[IDX], R0
                                CMPW (R0), #-1
                                BEQL 3$
                                CMPW (R0), CODE
                                BNEQ 2$
                                ADDL2 #2, R0
                                CVTWL (R0), R3
                                ADDL3 R0, R3, @RTXTC
                                MOVL #1, R0
                                RET
                                INCL IDX
                                BRB 1$
                                CLRL R0
                                RET
                                2$:
                                3$:
                                0C BC 00000000' 00 9E 00002
                                52 08 AC D0 0000A
                                51 D4 0000E
                                6241 DE 00010 1$:
                                FFFF 8F 60 B1 00014
                                19 13 00019
                                04 AC 60 B1 0001B
                                0F 12 0001F
                                50 02 C0 00021
                                53 60 32 00024
                                53 50 C1 00027
                                50 01 D0 0002C
                                04 0002F
                                51 D6 00030 2$:
                                DC 11 00032
                                50 D4 00034 3$:
                                04 00036
```

; Routine Size: 55 bytes, Routine Base: \$CODE\$ + 0DD1

.EXTRN LIB\$SIGNAL, LIB\$STOP

PSECT SUMMARY

Name	Bytes	Attributes						
\$PLITS	588	NOVEC,NOWRT,	RD	NOEXE,NOSHR,	LCL,	REL,	CON,NOPIC,ALIGN(2)	
\$GLOBALS	1288	NOVEC, WRT,	RD	NOEXE,NOSHR,	LCL,	REL,	CON,NOPIC,ALIGN(2)	
\$OWNS	332	NOVEC, WRT,	RD	NOEXE,NOSHR,	LCL,	REL,	CON,NOPIC,ALIGN(2)	
\$CODE\$	3592	NOVEC,NOWRT,	RD	EXE,NOSHR,	LCL,	REL,	CON,NOPIC,ALIGN(2)	

Library Statistics

File	----- Symbols -----		Pages Mapped	Processing Time
	Total	Loaded Percent		
\$255\$DUA28:[SYSLIB]STARLET.L32;1	9776	22 0	581	00:01.0
\$255\$DUA28:[NCP.OBJ]NMALIBRY.L32;1	887	22 2	47	00:00.7
\$255\$DUA28:[NCP.OBJ]NCPLIBRY.L32;1	373	22 5	52	00:00.3

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LI\$:NCPNETIO/OBJ=OBJ\$:NCPNETIO MSRC\$:NCPNETIO/UPDATE=(ENH\$:NCPNETIO)

Size: 3592 code + 2208 data bytes
Run Time: 00:54.9
Elapsed Time: 02:37.2
Lines/CPU Min: 2273
Lexemes/CPU-Min: 15466
Memory Used: 267 pages
Compilation Complete

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

0268 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

