

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

_S

Ps

NP

NP

\$G

\$O

NP

PA

_L

```

LL          IIIII
LL          IIIII
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LLLLLLLLLLL IIIII
LLLLLLLLLLL IIIII

SSSSSSSSS
SSSSSSSSS
SS
SS
SS
SS
SSSSSS
SSSSSS
SS
SS
SS
SS
SSSSSSSSS
SSSSSSSSS

```

```
0001 0 %TITLE 'Node File Routines for Network Management'
0002 0 MODULE NML$UPGRADE (
0003 0     LANGUAGE (BLISS32),
0004 0     ADDRESSING_MODE (NONEXTERNAL=GENERAL),
0005 0     ADDRESSING_MODE (EXTERNAL=GENERAL),
0006 0     IDENT = 'V04-000'
0007 0 ) =
0008 1 BEGIN
0009 1
0010 1
0011 1 *****
0012 1 *
0013 1 *  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0014 1 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0015 1 *  ALL RIGHTS RESERVED.
0016 1 *
0017 1 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0018 1 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0019 1 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0020 1 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0021 1 *  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0022 1 *  TRANSFERRED.
0023 1 *
0024 1 *  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0025 1 *  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0026 1 *  CORPORATION.
0027 1 *
0028 1 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0029 1 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0030 1 *
0031 1 *
0032 1 *****
0033 1
0034 1
0035 1 ++
0036 1 FACILITY:      DECnet Network Management Listener (NML)
0037 1
0038 1 ABSTRACT:
0039 1
0040 1     This module contains routines which are called only during the
0041 1     V4.0 upgrade procedure. The purpose of the routines is
0042 1     a) to convert the node database to a faster format
0043 1     utilizing 4 ISAM keys.
0044 1     b) since we are now marketing areas, to scan all
0045 1     the permanent database files which contain node numbers
0046 1     and make sure that any with an area number of 0 are
0047 1     given an area number. The area number used will be
0048 1     either one supplied by the customer during the upgrade,
0049 1     or area 1.
0050 1
0051 1 ENVIRONMENT:  VAX/VMS Operating System
0052 1
0053 1 AUTHOR:      Kathy Perko      , CREATION DATE: 31-Mar-1984
0054 1
0055 1 MODIFIED BY:
0056 1
0057 1 --
```

```
59 0058 1 %SBTTL 'Definitions'
60 0059 1
61 0060 1
62 0061 1 : TABLE OF CONTENTS:
63 0062 1 :
64 0063 1
65 0064 1 FORWARD ROUTINE
66 0065 1     NML$UPGRADE_PERM_DBS,
67 0066 1     NML_CONVERT_NODE_DB,
68 0067 1
69 0068 1     NML_OPEN_OLD_FILE,
70 0069 1
71 0070 1     NML_FIX_AREA_NUMS: NOVALUE,
72 0071 1
73 0072 1     NML_FIX_LOGGING_REC,
74 0073 1
75 0074 1     NML_FIX_NODE_PARAM;
76 0075 1
77 0076 1 : INCLUDE FILES:
78 0077 1 :
79 0078 1
80 0079 1 LIBRARY 'LIB$:NMLLIB.L32';
81 0080 1 LIBRARY 'SHRLIB$:NMLIBRY.L32';
82 0081 1 LIBRARY 'SYS$LIBRARY:STARLET.L32';
83 0082 1
84 0083 1 :
85 0084 1 : OWN STORAGE:
86 0085 1 :
87 0086 1 OWN
88 0087 1     nml$l_area: WORD;
89 0088 1
90 0089 1 :
91 0090 1 : Declare common NML external references.
92 0091 1 :
93 0092 1 $nml_extdef;
94 0093 1
95 0094 1 EXTERNAL
96 0095 1     nml$gq_node_file_dsc: VECTOR [2];
97 0096 1
98 0097 1 EXTERNAL LITERAL
99 0098 1     nml$_nodcvterr;
100 0099 1
101 0100 1 EXTERNAL ROUTINE
102 0101 1     lib$cvt_dtb,
103 0102 1     lib$get_symbol,
104 0103 1     nma$closefile,
105 0104 1     nma$deletefld,
106 0105 1     nma$openfile,
107 0106 1     nma$searchfld,
108 0107 1     nma$selectfile,
109 0108 1     nml$create_node_db,
110 0109 1     nml$connect_node_rab,
111 0110 1     nml$bld_reply,
112 0111 1     nml$getnxtsrc,
113 0112 1     nml$logfileop,
114 0113 1     nml$logrecordop,
115 0114 1     nml$matchrecord,
```

```
: Upgrade to V4.0 permanent databases.
: Convert node database from single
:   key ISAM to four keyed ISAM.
: Open old node database file for
:   conversion to 4 keyed format.
: Remove area 0 from node numbers in
:   permanent databases.
: Remove area 0 from a logging database
:   record.
: Remove area 0 from a node parameter.
```

NML\$UPGRADE
V04-000

Node File Routines for Network Management
Definitions

E 9
16-Sep-1984 00:37:14
14-Sep-1984 12:50:21

VAX-11 Bliss-32 V4.0-742
[NML.SRC]NMLUPGRAD.B32;1

Page 3
(2)

:	116	0115	1	nml\$openfile,
:	117	0116	1	nml\$readrecord,
:	118	0117	1	nml\$write_node_rec,
:	119	0118	1	nml\$write_record;
:	120	0119	1	

```
122 0120 1 %SBTTL 'nml$upgrade_perm_dbs Do V4.0 conversions on permanent db'
123 0121 1 GLOBAL ROUTINE nml$upgrade_perm_dbs =
124 0122 1
125 0123 1 !++
126 0124 1 FUNCTIONAL DESCRIPTION:
127 0125 1 This routine is called when doing a V4.0 upgrade. The permanent
128 0126 1 database files must be upgraded in two ways:
129 0127 1 a) convert the node database to a faster format
130 0128 1 utilizing 4 ISAM keys.
131 0129 1 b) since we are now marketing areas, scan all
132 0130 1 the permanent database files which contain node numbers
133 0131 1 and make sure that any with an area number of 0 are
134 0132 1 given an area number. The area number used will be
135 0133 1 either one supplied by the customer during the upgrade,
136 0134 1 or area 1. This change involves the following files
137 0135 1 and databases:
138 0136 1
139 0137 1 File Database entity Parameters
140 0138 1 ----
141 0139 1 NETLOGGING.DAT NML$C_LOGGING sink node
142 0140 1 source node
143 0141 1 NETX25.DAT NML$C_X25_SERV_DEST node
144 0142 1 NML$C_X25_ACCESS node
145 0143 1 NETX29.DAT NML$C_S29_SERV_DEST node
146 0144 1
147 0145 1 FORMAL PARAMETERS:
148 0146 1 None
149 0147 1
150 0148 1 ROUTINE VALUE:
151 0149 1 COMPLETION CODES:
152 0150 1 Most errors are signalled by subroutines called.
153 0151 1
154 0152 1 --
155 0153 1
156 0154 2 BEGIN
157 0155 2
158 0156 2 LOCAL
159 0157 2 status,
160 0158 2 area_dsc: VECTOR [2],
161 0159 2 ascii_area: BBLOCK [16];
162 0160 2
163 0161 2
164 0162 2 Get the area symbol set up when interrogating the customer for the upgrade.
165 0163 2 This symbol specifies the area number the customer wishes to convert his
166 0164 2 network to. It defaults to 1.
167 0165 2
168 0166 2 area_dsc [0] = 16;
169 0167 2 area_dsc [1] = ascii_area;
170 0168 2 status = LIB$GET_SYMBOL (UPLIT (%CHARCOUNT ('NET$AREA_NUM')),
171 0169 2 UPLIT BYTE ('NET$AREA_NUM')),
172 0170 2 area_dsc,
173 0171 2 area_dsc [0]);
174 0172 2 IF .status THEN
175 0173 2 status = LIB$CVT_DTB (.area_dsc [0],
176 0174 2 .area_dsc [1],
177 0175 2 nml$l_area);
178 0176 2 IF NOT .status THEN
```

```
: 179      0177 2      nml$sl_area = 1;
: 180      0178 2      |
: 181      0179 2      | Convert area 0 in the logging, and X25 and X29 Server Destination databases.
: 182      0180 2      |
: 183      0181 2      | nml_fix_area_nums (nml$sc_logging);
: 184      0182 2      | nml_fix_area_nums (nml$sc_x25_serv_dest);
: 185      0183 2      | nml_fix_area_nums (nml$sc_x25_access);
: 186      0184 2      | nml_fix_area_nums (nml$sc_x29_serv_dest);
: 187      0185 2      |
: 188      0186 2      | Convert the node database to use 4 ISAM keys. This makes it much faster.
: 189      0187 2      |
: 190      0188 2      | status = nml_convert_node_db ();
: 191      0189 2      | RETURN .status;
: 192      0190 1      | END;                                ! End of NML_UPGRADE_PERM_DBS
```

```
.TITLE NML$UPGRADE Node File Routines for Network Mana
      gement
```

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

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

```
4D 55 4E 5F 41 45 52 41 24 54 45 4E 00000 P.AAB:
      0000000C 0000C P.AAA:
      00000000' 00010
```

```
.ASCII \NET$AREA_NUM\
```

```
.LONG 12
```

```
.ADDRESS P.AAB
```

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

```
00000 NML$SL_AREA:
```

```
.BLKB 2
```

```
.EXTRN NML$GB_EVTSRCTYP
```

```
.EXTRN NML$GQ_EVTSRCDS
```

```
.EXTRN NML$GW_EVTCLASS
```

```
.EXTRN NML$GB_EVTMSKTYP
```

```
.EXTRN NML$GQ_EVTMSKDS
```

```
.EXTRN NML$GW_EVTSNKADR
```

```
.EXTRN NML$GW_ACP_CHAN
```

```
.EXTRN NML$GL_LOGMASK, NML$GQ_ENTSTRDSC
```

```
.EXTRN NML$AB_QIOBUFFER
```

```
.EXTRN NML$GQ_QIOBFDSC
```

```
.EXTRN NML$AB_EXEBUFFER
```

```
.EXTRN NML$GL_EXEDATPTR
```

```
.EXTRN NML$GQ_EXEDATDSC
```

```
.EXTRN NML$GQ_EXEBFDSC
```

```
.EXTRN NML$AB_RCVBUFFER
```

```
.EXTRN NML$GQ_RCVBFDSC
```

```
.EXTRN NML$AB_SNDBUFFER
```

```
.EXTRN NML$GQ_SNDBFDSC
```

```
.EXTRN NML$GL_RCVDATLEN
```

```
.EXTRN NML$AB_CPTABLE, NML$AB_MSGBLOCK
```

```
.EXTRN NML$AB_ENTITY_ID
```

```
.EXTRN NML$AB_QUALIFIER_ID
```

```
.EXTRN NML$AB_ENTITYDATA
```

```
.EXTRN NML$AB_NML_NMV, NML$AB_PRMSM
```

```
.EXTRN NML$AB_RECBUF, NML$AL_ENTINF
```

```
.EXTRN NML$AL_PERMINF
```

Address	Hex Data	ASCII Data
54	00000000'	00 001C 00000
53	00000000v	00 9E 00002
5E		00 9E 00009
10 AE		18 C2 00010
14 AE		10 D0 00013
		6E 9E 00017
	10 AE	9F 0001B
	14 AE	9F 0001E
00000000G	00000000'	00 9F 00021
		03 FB 00027
		50 D0 0002E
		52 E9 00031
		54 DD 00034
	18 AE	DD 00036
	18 AE	DD 00039
00000000G	00	03 FB 0003C
	52	50 D0 00043
	03	52 E8 00046
	64	01 B0 00049
		01 DD 0004C
	63	01 FB 0004E
		12 DD 00051
	63	01 FB 00053
		0D DD 00056
	63	01 FB 00058
		16 DD 0005B
	63	01 FB 0005D
00000000v	00	00 FB 00060
	52	50 D0 00067

```

. EXTRN  NML$AW_PRM_DES, NML$GB_CMD_VER
. EXTRN  NML$GB_ENTITY_CODE
. EXTRN  NML$GB_ENTITY_FORMAT
. EXTRN  NML$GL_QUALIFIER_PST
. EXTRN  NML$GB_QUALIFIER_FORMAT
. EXTRN  NML$GB_FUNCTION
. EXTRN  NML$GB_INFO, NML$GB_OPTIONS
. EXTRN  NML$GL_PRCODE, NML$GL_PRS_FLGS
. EXTRN  NML$GL_NML_ENTITY
. EXTRN  NML$GQ_NETNAMDSC
. EXTRN  NML$GQ_RECBF DSC
. EXTRN  NML$GW_PRMDESCNT
. EXTRN  NML$GQ_NODE_FILE_DSC
. EXTRN  NML$NODCVTERR, LIB$CVT_DTB
. EXTRN  LIB$GET_SYMBOL, NMA$CLOSEFILE
. EXTRN  NMA$DELETEFLD, NMA$OPENFILE
. EXTRN  NMA$SEARCHFLD, NMA$SELECTFILE
. EXTRN  NML$CREATE_NODE_DB
. EXTRN  NML$CONNECT_NODE_RAB
. EXTRN  NML$BLD_REPLY, NML$GETNXTSRC
. EXTRN  NML$LOGFILEOP, NML$LOGRECORDOP
. EXTRN  NML$MATCHRECORD
. EXTRN  NML$OPENFILE, NML$READRECORD
. EXTRN  NML$WRITE_NODE_REC
. EXTRN  NML$WRITERECORD

```

.PSECT \$CODES,NOWRT,2

```

.ENTRY    NML$UPGRADE_PERM_DBS, Save R2,R3,R4
MOVAB     NML$SL_AREA, R4
MOVAB     NML_FIX_AREA_NUMS, R3
SUBL2     #24, SP
MOVL      #16, AREA_DSC
MOVAB     ASCII_AREA, AREA_DSC+4
PUSHAB    AREA_DSC
PUSHAB    AREA_DSC
PUSHAB    P.AAA
CALLS     #3, LIB$GET_SYMBOL
MOVL      R0, STATUS
BLBC      STATUS, 1$
PUSHL     R4
PUSHL     AREA_DSC+4
PUSHL     AREA_DSC
CALLS     #3, LIB$CVT_DTB
MOVL      R0, STATUS
BLBS      STATUS, 2$
MOVW      #1, NML$SL_AREA
PUSHL     #1
CALLS     #1, NML_FIX_AREA_NUMS
PUSHL     #18
CALLS     #1, NML_FIX_AREA_NUMS
PUSHL     #13
CALLS     #1, NML_FIX_AREA_NUMS
PUSHL     #22
CALLS     #1, NML_FIX_AREA_NUMS
CALLS     #0, NML_CONVERT_NODE_DB
MOVL      R0, STATUS

```

0121
0166
0167
0171
0168
0172
0173
0174
0173
0176
0177
0181
0182
0183
0184
0188

NML\$UPGRADE
V04-000

Node File Routines for Network Management
nml\$upgrade_perm_dbs Do V4.0 conversions on per

1 9
16-Sep-1984 00:37:14
14-Sep-1984 12:50:21

VAX-11 Bliss-32 V4.0-742
[NML.SRC]NMLUPGRAD.B32;1

Page 7
(3)

04 0006A

RET

; 0190

; Routine Size: 107 bytes, Routine Base: \$CODE\$ + 0000

```
194 0191 1 %SBTTL 'nml_convert_node_db          Convert node permanent database file'
195 0192 1 ROUTINE nml_convert_node_db =
196 0193 1
197 0194 1 ++
198 0195 1 FUNCTIONAL DESCRIPTION:
199 0196 1     This routine is called if the node permanent database is opened and
200 0197 1     only has 1 ISAM key. The old node database file had only 1 ISAM key
201 0198 1     and was too slow with large numbers of nodes in the database.
202 0199 1     This routine converts the old node database file to the new node
203 0200 1     database format using multiple ISAM keys. It's much faster to access.
204 0201 1
205 0202 1 FORMAL PARAMETERS:
206 0203 1     None
207 0204 1
208 0205 1 ROUTINE VALUE:
209 0206 1 COMPLETION CODES:
210 0207 1
211 0208 1     Failure or RMS error
212 0209 1
213 0210 1 --
214 0211 1
215 0212 2 BEGIN
216 0213 2
217 0214 2 OWN
218 0215 2     temp_node_file_dsc : VECTOR [2]
219 0216 2         INITIAL (%CHARCOUNT ('TEMP_NETNODE'),
220 0217 2             UPLIT BYTE ('TEMP_NETNODE'));
221 0218 2
222 0219 2 LOCAL
223 0220 2     status,
224 0221 2     new_netnode_fab:    $FAB DECL,
225 0222 2     old_rab_addr:      REF BBLOCK,      ! Address of RAB for old node database
226 0223 2                                     ! file.
227 0224 2     old_rec_dsc:       VECTOR [2],      ! Descriptor of record from old file
228 0225 2     new_rec:           BBLOCK [nm($k_recbflen),
229 0226 2     new_rec_dsc:       VECTOR [2],
230 0227 2     field_size,
231 0228 2     field_addr,
232 0229 2     params_len,
233 0230 2     node_type,
234 0231 2     key_value_dsc:     VECTOR [2],
235 0232 2     fab,
236 0233 2     msgsize;           ! Size of signalled error message.
237 0234 2
238 0235 2
239 0236 2     Reopen old node permanent database file for sequential read access.
240 0237 2
241 0238 2     status = nml_open_old_file (old_rab_addr);
242 0239 2
243 0240 2     If the old node database was opened successfull, then create the new node
244 0241 2     database file with multiple ISAM keys.
245 0242 2
246 0243 2 IF .status THEN
247 0244 2     status = nml$create_node_db (temp_node_file_dsc, fab);
248 0245 2 IF .status THEN
249 0246 2     status = nml$connect_node_rab ();
250 0247 2 IF NOT .status THEN
```

```
251 0248 2 RETURN .status;
252 0249 2
253 0250 2 Build new node database file from old node database file.
254 0251 2
255 0252 2 new_rec_dsc [1] = new_rec + nml$node_keys_len;
256 0253 2 old_rec_dsc [1] = .nm[$gq_recbfdsc [dsc$a_pointer] + nml$perm_keys_len;
257 0254 2 key_value_dsc [1] = new_rec [nmn$w_key_add];
258 0255 2 DO
259 0256 2 BEGIN
260 0257 2 Read a record from the old node permanent database.
261 0258 2
262 0259 2 status = $GET (RAB = .old_rab_addr);
263 0260 2 IF .status THEN
264 0261 2 BEGIN
265 0262 2 old_rec_dsc [0] = .old_rab_addr [rab$w_rsz] - nml$perm_keys_len;
266 0263 2
267 0264 2 The two of the keys for the new file are the node address and name.
268 0265 2 Both these keys are also NICE parameters. The routine that writes
269 0266 2 the record to the node file fixes the key values for these two
270 0267 2 keys.
271 0268 2
272 0269 2 Another key in the new node database is the node type. The
273 0270 2 three node types are executor node, remote node, and loop node.
274 0271 2 The write routine must be given the node entity type to determine
275 0272 2 this key value, since it isn't a NICE parameter.
276 0273 2
277 0274 2 field_addr = 0;
278 0275 2 IF nma$searchfld (old_rec_dsc,
279 0276 2 nml$sc_key_exe,
280 0277 2 field_size,
281 0278 2 field_addr) THEN
282 0279 2 BEGIN
283 0280 2 node_type = nml$sc_executor;
284 0281 2
285 0282 2 Get rid of the executor's OWNER field from the parameters,
286 0283 2 since it isn't a real NICE parameter. (For the other node
287 0284 2 records, the OWNER is a valid NICE parameter.)
288 0285 2
289 0286 2 nma$deletefld (old_rec_dsc, nml$sc_key_exe);
290 0287 2 END
291 0288 2 ELSE
292 0289 2 BEGIN
293 0290 2 node_type = nml$sc_node;
294 0291 2
295 0292 2 If there is not node address and it's got a circuit parameter (NLI),
296 0293 2 then it's a loop node.
297 0294 2
298 0295 2 field_addr = 0;
299 0296 2 status = nma$searchfld (old_rec_dsc,
300 0297 2 nml$sc_pcho_add,
301 0298 2 key_value_dsc [0],
302 0299 2 field_addr);
303 0300 2
304 0301 2 If there is no address for the node, it must be a loopnode.
305 0302 2 Loopnodes always have a circuit (NLI) parameter.
306 0303 2
307 0304 2
```

```
308      IF (NOT .status) AND
309          (status = nma$searchfld (old_rec_dsc,
310              nma$pcno_nli,
311              field_size,
312              field_addr)) THEN
313          node_type = nml$loopnode;
314      IF NOT .status THEN
315          RETURN .status;
316      END;
317      Move the node parameters from the old record to the new one, putting
318      them right behind the keys.
319      CH$MOVE (.old_rec_dsc [0], .old_rec_dsc [1],
320          new_rec [nmn$a_nod_params]);
321      The record descriptors to not include the keys. So the new record
322      length is the same as the old record's length - the data length.
323      new_rec_dsc [0] = .old_rec_dsc [0];
324      If node numbers are in area 0, fix them.
325      nml_fix_node_param (nma$pcno_add, new_rec_dsc);
326      nml_fix_node_param (nma$pcno_iho, new_rec_dsc);
327      Write the record to the new node database file.
328      status = nml$write_node_rec ( nmn$put_rec,      ! Add a new node
329          .node_type,      ! for node type key
330          new_rec_dsc);      ! Record to write
331      END;
332      END
333      UNTIL NOT .status;
334      Close the old permanent database node file (may it rest in peace).
335      nma$closefile (nma$opn_node);
336      Now, close the new temporary node file, rename it to it's permanent name,
337      and delete any old versions of the temporary node file that may have
338      been left around if the system crashed while the conversion was being
339      done.
340      status = $CLOSE (FAB = .fab);
341      $FAB_INIT (FAB = new_netnode_fab,
342          DNM = 'SYSS$SYSTEM:.DAT',
343          FNA = .nml$gq_node_file_dsc [1],
344          FNS = .nml$gq_node_file_dsc [0]
345      );
346      IF .status THEN
347          status = $RENAME (OLDFAB = .fab,
348              NEWFAB = new_netnode_fab);
349      IF .status EQL rms$suc THEN
350          nml$logfileop (dbg$sc_fileio,
351              nma$opn_node,
352              $ASCII ('file converted'));
353      P
354      P
355      P
356      P
357      P
358      P
359      P
360      P
361      P
362      P
363      P
364      P
```

nml_convert_node_db

[illegible]

00000000G	00	02	FB	00034	CALLS	#2, NML\$CREATE_NODE_DB	:			
	56	50	D0	0003B	MOVL	R0, STATUS	:			
	0A	56	E9	0003E	BLBC	STATUS, 1\$	0245			
00000000G	00	00	FB	00041	CALLS	#0, NML\$CONNECT_NODE_RAB	0246			
	56	50	D0	00048	MOVL	R0, STATUS	:			
	03	56	E8	0004B	BLBS	STATUS, 2\$	0247			
		0092	31	0004E	BRW	6\$	:			
20	AE	2E	AE	9E	00051	2\$: MOVAB	NEW_REC+10, NEW_REC_DSC+4	0252		
AC AD 00000000G	00	02	C1	00056	ADDL3	#2, NML\$GQ RECBFDSC+4, OLD_REC_DSC+4	0253			
18	AE	24	AE	9E	0005F	MOVAB	NEW_REC, KEY_VALUE_DSC+4	0254		
	57	6E	D0	00064	MOVL	OLD_RAB_ADDR, R7	0260			
		57	DD	00067	3\$: PUSHL	R7	:			
00000000G	00	01	FB	00069	CALLS	#1, SYSSGET	:			
	56	50	D0	00070	MOVL	R0, STATUS	:			
	03	56	E8	00073	BLBS	STATUS, 4\$	0261			
		00A6	31	00076	BRW	8\$	:			
A8	AD	22	A7	3C	00079	4\$: MOVZWL	34(R7), OLD_REC_DSC	0263		
A8	AD		02	C2	0007E	SUBL2	#2, OLD_REC_DSC	:		
		08	AE	D4	00082	CLRL	FIELD_ADDR	0275		
		08	AE	9F	00085	PUSHAB	FIELD_ADDR	0276		
		10	AE	9F	00088	PUSHAB	FIELD_SIZE	:		
	7E		03	CE	0008B	MNEGL	#3, -(SP)	:		
		A8	AD	9F	0008E	PUSHAB	OLD_REC_DSC	:		
	6A		04	FB	00091	CALLS	#4, NMA\$SEARCHFLD	:		
	12		50	E9	00094	BLBC	R0, 5\$	:		
	58		07	D0	00097	MOVL	#7, NODE_TYPE	0281		
	7E		03	CE	0009A	MNEGL	#3, -(SP)	0287		
		A8	AD	9F	0009D	PUSHAB	OLD_REC_DSC	:		
00000000G	00	02	FB	000A0	CALLS	#2, NMA\$DELETEFLD	:			
		3E	11	000A7	BRB	7\$	0276			
	58	03	D0	000A9	5\$: MOVL	#3, NODE_TYPE	0291			
		08	AE	D4	000AC	CLRL	FIELD_ADDR	0296		
		08	AE	9F	000AF	PUSHAB	FIELD_ADDR	0297		
		18	AE	9F	000B2	PUSHAB	KEY_VALUE_DSC	0299		
	7E	01F6	8F	3C	000B5	MOVZWL	#502, -(SP)	0297		
		A8	AD	9F	000BA	PUSHAB	OLD_REC_DSC	:		
	6A		04	FB	000BD	CALLS	#4, NMA\$SEARCHFLD	:		
	56		50	D0	000C0	MOVL	R0, STATUS	:		
	21		56	E8	000C3	BLBS	STATUS, 7\$	0305		
		08	AE	9F	000C6	PUSHAB	FIELD_ADDR	0306		
		10	AE	9F	000C9	PUSHAB	FIELD_SIZE	:		
	7E	01F5	8F	3C	000CC	MOVZWL	#501, -(SP)	:		
		A8	AD	9F	000D1	PUSHAB	OLD_REC_DSC	:		
	6A		04	FB	000D4	CALLS	#4, NMA\$SEARCHFLD	:		
	56		50	D0	000D7	MOVL	R0, STATUS	:		
	06		56	E9	000DA	BLBC	STATUS, 6\$	:		
	58		05	D0	000DD	MOVL	#5, NODE_TYPE	0310		
	04		56	E8	000E0	BLBS	STATUS, 7\$	0311		
	50		56	D0	000E3	6\$: MOVL	STATUS, R0	0312		
			04	000E6	RET		:	:		
2E	AE	AC	BD	A8	AD	28	000E7	7\$: MOVCL3	OLD_REC_DSC, @OLD_REC_DSC+4, NEW_REC+10	0319
		1C	AE	A8	AD	D0	000EE	MOVL	OLD_REC_DSC, NEW_REC_DSC	0324
				1C	AE	9F	000F3	PUSHAB	NEW_REC_DSC	0328
	7E		01F6	8F	3C	000F6	MOVZWL	#502, -(SP)	:	:
	6B			02	FB	000FB	CALLS	#2, NML_FIX_NODE_PARAM	:	:
				1C	AE	9F	000FE	PUSHAB	NEW_REC_DSC	0329
	7E		8D	8F	9A	00101	MOVZBL	#14T, -(SP)	:	:

0050	8F	00	6B	02	FB	00105	CALLS	#2, NML_FIX_NODE_PARAM		
			1C	AE	9F	00108	PUSHAB	NEW_REC_DSC	0333	
				58	DD	0010B	PUSHL	NODE_TYPE	0334	
				01	DD	0010D	PUSHL	#1	0333	
		00000000G	00	03	FB	0010F	CALLS	#3, NML\$WRITE_NODE_REC		
			56	50	DO	00116	MOVL	R0, STATUS		
			03	56	E9	00119	BLBC	STATUS, 8\$	0338	
				FF48	31	0011C	BRW	3\$		
				7E	D4	0011F	CLRL	-(SP)	0342	
		00000000CG	00	01	FB	00121	CALLS	#1, NMA\$CLOSEFILE		
				AE	DD	00128	PUSHL	FAB	0349	
		00000000G	00	01	FB	0012B	CALLS	#1, SYSS\$CLOSE		
			56	50	DO	00132	MOVL	R0, STATUS		
			6E	00	2C	00135	MOVCS	#0, (SP), #0, #80, \$RMS_PTR	0354	
				B0	AD	0013C				
		B0	AD	5003	8F	B0	0013E	MOVW	#20483, \$RMS_PTR	
		C6	AD		02	90	00144	MOVB	#2, \$RMS_PTR+22	
		CF	AD		02	90	00148	MOVB	#2, \$RMS_PTR+31	
		DC	AD	00000000G	00	DO	0014C	MOVL	NML\$GQ_NODE_FILE_DSC+4, \$RMS_PTR+44	
		E0	AD	00000000'	00	9E	00154	MOVAB	P.AAD, \$RMS_PTR+48	
		E4	AD	00000000G	00	90	0015C	MOVB	NML\$GQ_NODE_FILE_DSC, \$RMS_PTR+52	
		E5	AD		0F	90	00164	MOVB	#15, \$RMS_PTR+53	
			12	56	E9	00168	BLBC	STATUS, 9\$	0355	
				B0	AD	9F	0016B	PUSHAB	NEW_NETNODE_FAB	0357
				7E	7C	0016E	CLRQ	-(SP)		
				10	AE	DD	00170	PUSHL	FAB	
		00000000G	00	04	FB	00173	CALLS	#4, SYSS\$RENAME		
			56	50	DO	0017A	MOVL	R0, STATUS		
		00010001	8F	56	D1	0017D	CMPL	STATUS, #65537	0358	
				10	12	00184	BNEQ	10\$		
				00000000'	00	9F	00186	PUSHAB	P.AAE	0361
			7E	01	7D	0018C	MOVQ	#1, -(SP)	0359	
		00000000G	00	03	FB	0018F	CALLS	#3, NML\$LOGFILEOP		
			39	56	E8	00196	BLBS	STATUS, 11\$	0362	
			69	8F	9A	00199	MOVZBL	#78, NML\$AB_MSGBLOCK	0369	
		04	A9	12	8E	0019D	MNEGB	#18, NML\$AB_MSGBLOCK+4	0371	
				08	A9	B4	001A1	CLRW	NML\$AB_MSGBLOCK+8	0372
		0C	A9	00000000G	8F	DO	001A4	MOVL	#NML\$NODCVTERR, NML\$AB_MSGBLOCK+12	0373
			10	56	DO	001AC	MOVL	STATUS, NML\$AB_MSGBLOCK+16	0374	
				10	AE	9F	001B0	PUSHAB	MSGSIZE	0376
				59	DD	001B3	PUSHL	R9		
		00000000G	00	02	FB	001B5	CALLS	#2, NML\$BLD_REPLY		
				10	AE	DD	001BC	PUSHL	MSGSIZE	0377
				00000000G	00	9F	001BF	PUSHAB	NML\$AB_SNDBUFFER	
				01F90000	8F	DD	001C5	PUSHL	#33095680	
		00000000G	00	03	FB	001CB	CALLS	#3, LIB\$SIGNAL		
			50	01	DO	001D2	MOVL	#1, R0	0381	
				04	001D5	RET			0382	

; Routine Size: 470 bytes, Routine Base: \$CODE\$ + 006B

```
387 0383 1 %SBTTL 'nml_open_old_file      Open old node database for conversion'
388 0384 1 ROUTINE nml_open_old_file (rab_addr) =
389 0385 1
390 0386 1 !++
391 0387 1 FUNCTIONAL DESCRIPTION:
392 0388 1     This routine is called when creating a new node database file. It
393 0389 1     opens the old node database file for sequential access.
394 0390 1
395 0391 1 FORMAL PARAMETERS:
396 0392 1     rab_addr      Address of RAB returned for use when reading records
397 0393 1                  from the file.
398 0394 1
399 0395 1 ROUTINE VALUE:
400 0396 1 COMPLETION CODES:
401 0397 1     Failure or RMS error
402 0398 1
403 0399 1 --
404 0400 1
405 0401 2 BEGIN
406 0402 2
407 0403 2 LOCAL
408 0404 2     rab:          REF BBLOCK,
409 0405 2     status,
410 0406 2     file_dsc:     REF BBLOCK;
411 0407 2
412 0408 2 status = nma$openfile (nma$c_opn_node, nma$c_opn_ac_ro);
413 0409 2
414 0410 2 If the old node database file was successfully opened, change
415 0411 2 the RAB to get the file's records sequentially.
416 0412 2
417 0413 2 IF .status THEN
418 0414 2 BEGIN
419 0415 2     status = nma$selectfile (nma$c_opn_node, file_dsc);
420 0416 2     rab = .file_dsc [12, 0, 32, 0];
421 0417 2     rab [rab$b_rac] = rab$c_seq;
422 0418 2
423 0419 2     Set up the RABs input buffer fields.
424 0420 2
425 0421 2     rab [rab$w_usz] = .nml$gq_recbfdsc [dsc$w_length];
426 0422 2     rab [rab$l_ubf] = .nml$gq_recbfdsc [dsc$a_pointer];
427 0423 2
428 0424 2     Return RAB address to calling routine.
429 0425 2
430 0426 2     .rab_addr = .rab;
431 0427 2 END;
432 0428 2 RETURN .status;
433 0429 1 END;      ! of      nml_open_old_file
```

0000 00000 NML_OPEN_OLD FILE:

5E
00000000G 0004 C2 00002
7E 7C 00005
02 FB 00007-WORD Save nothing
SUBL2 #4, SP
CLRQ -(SP)
CALLS #2, NMA\$OPENFILE: 0384
: 0408
:

NML\$UPGRADE
V04-000

Node File Routines for Network Management
nml_open_old_file Open old node database for co

D 10
16-Sep-1984 00:37:14
14-Sep-1984 12:50:21

VAX-11 Bliss-32 V4.0-742
[NML.SRC]NMLUPGRAD.B32;1

Page 15
(5)

29			50	E9	0000E	BLBC	STATUS, 1\$	
			5E	DD	00011	PUSHL	SP	: 0413
			7E	D4	00013	CLRL	-(SP)	: 0415
00000000G	00		02	FB	00015	CALLS	#2, NMA\$SELECTFILE	
	51		6E	D0	0001C	MOVL	FILE_DSC, R1	: 0416
	51	0C	A1	D0	0001F	MOVL	12(RT), RAB	
		1E	A1	94	00023	CLRB	30(RAB)	: 0417
20	A1	00000000G	00	B0	00026	MOVW	NML\$GQ_RECBFDSC, 32(RAB)	: 0421
24	A1	00000000G	00	D0	0002E	MOVL	NML\$GQ_RECBFDSC+4, 36(RAB)	: 0422
04	BC		51	D0	00036	MOVL	RAB, @RAB_ADDR	: 0426
			04	0003A	1\$:	RET		: 0429

; Routine Size: 59 bytes, Routine Base: \$CODE\$ + 0241

```

435 0430 1 %SBTTL 'nml_fix_area_nums Convert permanent db area 0 to valid area'
436 0431 1 ROUTINE nml_fix_area_nums (entity) : NOVALUE =
437 0432 1
438 0433 1 !++
439 0434 1 ! FUNCTIONAL DESCRIPTION:
440 0435 1 ! This routine is called when doing a V4.0 upgrade. It scans all
441 0436 1 ! permanent database files (except the node database) containing node
442 0437 1 ! numbers. Each node number which has an area number of 0 is given
443 0438 1 ! an area number supplied by the customer or 1.
444 0439 1
445 0440 1 ! FORMAL PARAMETERS:
446 0441 1 ! entity NML entity type which has a node number in it's database.
447 0442 1 ! This parameter can be one of the following:
448 0443 1 ! nml$c_logging
449 0444 1 ! nml$c_x25_serv_dest
450 0445 1 ! nml$c_x29_serv_dest
451 0446 1
452 0447 1
453 0448 1 ! ROUTINE VALUE:
454 0449 1 ! COMPLETION CODES:
455 0450 1
456 0451 1 !--
457 0452 1
458 0453 2 BEGIN
459 0454 2
460 0455 2 LOCAL
461 0456 2 fid, ! File id of permanent database file
462 0457 2 owner, ! Entity record owner field id
463 0458 2 convert_routine, ! Conversion routine for entity
464 0459 2 param_id, ! NMA parameter id for node parameter to fix.
465 0460 2 recdsc: VECTOR [2], ! Descriptor of permanent database record
466 0461 2 ! being converted.
467 0462 2 key; ! Key of current file record.
468 0463 2
469 0464 2 SELECTONEU .entity OF
470 0465 2 SET
471 0466 2 [nml$c_logging]:
472 0467 3 BEGIN
473 0468 3 convert_routine = nml_fix_logging_rec;
474 0469 3 param_id = nma$c_pclo_sin;
475 0470 2 END;
476 0471 2 [nml$c_x25_serv_dest,
477 0472 2 nml$c_x29_serv_dest]:
478 0473 3 BEGIN
479 0474 3 convert_routine = nml_fix_node_param;
480 0475 3 param_id = nma$c_pcxs_nod;
481 0476 2 END;
482 0477 2 [nml$c_x25_access]:
483 0478 3 BEGIN
484 0479 3 convert_routine = nml_fix_node_param;
485 0480 3 param_id = nma$c_pcxa_nod;
486 0481 2 END;
487 0482 2 TES;
488 0483 2 fid = .nml$ab_entitydata [.entity, eit$b_fileid];
489 0484 2 owner = .nml$ab_entitydata [.entity, eit$w_key];
490 0485 2
491 0486 2 ! First open the permanent database file to convert. If it doesn't exist,
```

```
492 0487 2 ! there's nothing to do.
493 0488
494 0489 nml$openfile (.fid, nma$c_opn_ac_rw);
495 0490 key = 0;
496 0491
497 0492 Read a record for the entity. The logging and X25/9 Server Destination
498 0493 databases are both in database files which contain entries for several
499 0494 entities. The owner field distinguishes them.
500 0495
501 0496 WHILE nml$matchrecord (.fid,
502 0497 nml$gq_recbfdsc,
503 0498 key,
504 0499 .owner, 0, 0,
505 0500 0, 0, 0, ! No qualifier
506 0501 recdsc) DO
507 0502 BEGIN
508 0503
509 0504 Check the record to see if it contains any node addresss, and, if it
510 0505 does, convert them. The convert_routine will return success if it
511 0506 found any node addresses, in which case, write the updated record
512 0507 back to the permanent database file.
513 0508
514 0509 IF (.convert_routine) (.param_id, recdsc) THEN
515 0510 nml$writerecord (.fid, .entity, key, recdsc, 0);
516 0511 key = .key + 1;
517 0512 END;
518 0513 1 END; ! End of NML_FIX_AREA_NUMS
```

```
007C 00000 NML_FIX_AREA_NUMS:
5E 0C C2 00002 .WORD Save R2,R3,R4,R5,R6 : 0431
52 04 AC D0 00005 SUBL2 #12, SP
01 52 D1 00009 MOVL ENTITY, R2 : 0464
54 00000000V 00 9E 0000E BNEQ R2, #1 : 0466
53 C8 8F 9A 00015 MOVAB NML_FIX_LOGGING_REC, CONVERT_ROUTINE : 0468
12 1B 11 00019 MOVZBL #200, PARAM_ID : 0469
16 52 D1 0001B 1$: BRB 3$ : 0464
0D 0A 13 0001E CMPL R2, #18 : 0471
54 00000000V 00 9E 0002A 2$: BEQL 2$
53 0140 8F 3C 00031 BEQL 2$
52 2C C5 00036 3$: CMPL R2, #22
55 00000000G0040 9A 0003A BEQL 2$
56 00000000G0040 9F 00042 CMPL R2, #13 : 0477
9E 3C 00049 BNEQ 3$
01 DD 0004C MOVAB NML_FIX_NODE_PARAM, CONVERT_ROUTINE : 0479
55 DD 0004E MOVZWL #320, PARAM_ID : 0480
00000000G 00 02 FB 00050 MULL3 #44, R2, R0 : 0483
6E D4 00057 MOVZBL NML$AB_ENTITYDATA[R0], FID : 0484
PUSHAB NML$AB_ENTITYDATA+3[R0]
MOVZWL @ (SP)+, OWNER : 0489
PUSHL #1
PUSHL FID
CALLS #2, NML$OPENFILE : 0490
CLRL KEY
```

	04	AE 9F 00059 4\$:	PUSHAB RECDSC	: 0496
		7E 7C 0005C	CLRQ -(SP)	:
		7E 7C 0005E	CLRQ -(SP)	:
		7E D4 00060	CLRL -(SP)	:
		56 DD 00062	PUSHL OWNER	: 0499
	1C	AE 9F 00064	PUSHAB KEY	: 0496
	00000000G	00 9F 00067	PUSHAB NML\$GQ_REC BFDSC	:
		55 DD 0006D	PUSHL FID	:
00000000G	00	0A FB 0006F	CALLS #10, NML\$MATCHRECORD	:
	22	50 E9 00076	BLBC R0, 6\$	:
	04	AE 9F 00079	PUSHAB RECDSC	: 0509
		53 DD 0007C	PUSHL PARAM ID	:
	64	02 FB 0007E	CALLS #2, (CONVERT_ROUTINE)	:
	13	50 E9 00081	BLBC R0, 5\$	:
		7E D4 00084	CLRL -(SP)	: 0510
	08	AE 9F 00086	PUSHAB RECDSC	:
	08	AE 9F 00089	PUSHAB KEY	:
		52 DD 0008C	PUSHL R2	:
		55 DD 0008E	PUSHL FID	:
00000000G	00	05 FB 00090	CALLS #5, NML\$WRITERECORD	:
		6E D6 00097 5\$:	INCL KEY	: 0511
		BE 11 00099	BRB 4\$	: 0496
		04 0009B 6\$:	RET	: 0513

; Routine Size: 156 bytes, Routine Base: \$CODE\$ + 027C

```

520 0514 1 %SBTTL 'nml_fix_logging_rec Upgrade V4.0 logging database'
521 0515 1 ROUTINE nml_fix_logging_rec (param_id, recdsc) =
522 0516 1
523 0517 1 ++
524 0518 1 FUNCTIONAL DESCRIPTION:
525 0519 1 This routine is called as part of the V4.0 upgrade. It finds node
526 0520 1 numbers in the logging database, and, if they have an area number of
527 0521 1 0, converts them to an area number supplied by the customer during
528 0522 1 the upgrade. (Area 0 is for Phase III nodes only). Node numbers
529 0523 1 occur in the logging database in two places. As sink nodes and as
530 0524 1 event sources. Both fields are in th EFI (Event Filter) database.
531 0525 1
532 0526 1 INPUTS:
533 0527 1 PARAM_ID NMA id for node parameter to convert.
534 0528 1 RECDSC Address of descriptor of a logging database record.
535 0529 1 --
536 0530 1
537 0531 2 BEGIN
538 0532 2
539 0533 2 LOCAL
540 0534 2 fldlen,
541 0535 2 event_blk_dsc : VECTOR [2],
542 0536 2 source_blk : REF BBLOCK,
543 0537 2 node_addr_changed;
544 0538 2
545 0539 2
546 0540 2 Find the sink node in the database record. This should always succeed.
547 0541 2
548 0542 2 node_addr_changed = nml_fix_node_param (.param_id,
549 0543 2 .recdsc);
550 0544 2
551 0545 2 Find the event parameter. This should always succeed as well.
552 0546 2
553 0547 2 event_blk_dsc [1] = 0;
554 0548 2 IF nma$searchfld (.recdsc,
555 0549 2 nma$c_pclo eve
556 0550 2 event_blk_dsc [0],
557 0551 2 event_blk_dsc [1]) THEN
558 0552 2 BEGIN
559 0553 2 source_blk = 0;
560 0554 2
561 0555 2 Examine each of the source blocks in the event block. If any
562 0556 2 have a source type = node, check the node address for area 0.
563 0557 2 If it's area 0 and not the executor node, give the area number
564 0558 2 the customer requested.
565 0559 2
566 0560 3 WHILE nml$getnxtsrc (event_blk_dsc, source_blk) DO
567 0561 4 BEGIN
568 0562 4 IF .source_blk [src$b_srctype] EQL nma$c_ent_nod THEN
569 0563 5 BEGIN
570 0564 5 BIND
571 0565 5 source_node = source_blk [src$w_nodadr] : BBLOCK;
572 0566 5
573 0567 5 IF .source_node [nma$v_area] EQL 0 AND
574 0568 5 .source_node [nma$w_node] NEQ 0 THEN
575 0569 6 BEGIN
576 0570 6 source_node [nma$v_area] = .nml$l_area;
```

```
: 577      0571  6      node_addr_changed = true;
: 578      0572  5      END;
: 579      0573  4      END;
: 580      0574  3      END;
: 581      0575  2      END;
: 582      0576  2      RETURN .node_addr_changed;
: 583      0577  2      ! End of NML_FIX_LOGGING_REC
: 584      0578  1      END;
```

```
                                0004 00000 NML_FIX_LOGGING_REC:
                                .WORD  Save R2
                                5E      0C  C2 00002  SUBL2  #12, SP
                                7E      04  AC 7D 00005  MOVQ  PARAM_ID, -(SP)
                                00000000V 00  02  FB 00009  CALLS  #2, NML_FIX_NODE_PARAM
                                52      50  D0 00010  MOVL  R0, NODE_ADDR_CHANGED
                                08      08  AE D4 00013  CLRL  EVENT_BLK_DSC+4
                                08      08  AE 9F 00016  PUSHAB EVENT_BLK_DSC+4
                                08      08  AE 9F 00019  PUSHAB EVENT_BLK_DSC
                                7E      C9  8F 9A 0001C  MOVZBL #201, -(SP)
                                00000000G 00  08  AC DD 00020  PUSHL  RECDSC
                                34      04  FB 00023  CALLS  #4, NMASSEARCHFLD
                                50      50  E9 0002A  BLBC  R0, 2$
                                6E      D4 0002D  CLRL  SOURCE_BLK
                                5E      DD 0002F 1$:  PUSHL  SP
                                08      08  AE 9F 00031  PUSHAB EVENT_BLK_DSC
                                00000000G 00  02  FB 00034  CALLS  #2, NML$GETNXTSRC
                                23      50  E9 0003B  BLBC  R0, 2$
                                50      6E  D0 0003E  MOVL  SOURCE_BLK, R0
                                03      A0  95 00041  TSTB  3(R0)
                                FC      8F      05  A0  93 00046  BITB  5(R0), #252
                                04      E2  12 0004B  BNEQ  1$
                                05      A0  B5 0004D  TSTW  4(R0)
                                06      DD  13 00050  BEQL  1$
                                02 00000000' 00  F0 00052  INSV  NML$ AREA, #2, #6, 5(R0)
                                52      01  D0 0005C  MOVL  #1, NODE_ADDR_CHANGED
                                50      CE  11 0005F  BRB   1$
                                52      D0 00061 2$:  MOVL  NODE_ADDR_CHANGED, R0
                                04 00064  RET
                                : 0515
                                : 0542
                                : 0547
                                : 0551
                                : 0550
                                : 0548
                                : 0553
                                : 0560
                                : 0562
                                : 0567
                                : 0568
                                : 0570
                                : 0571
                                : 0560
                                : 0577
                                : 0578
```

; Routine Size: 101 bytes, Routine Base: \$CODE\$ + 0318

```

: 586 0579 1 %SBTTL 'nml_fix_node_param      Upgrade V4.0 node param in perm database'
: 587 0580 1 ROUTINE nml_fix_node_param (param_id, recdsc) =
: 588 0581 1
: 589 0582 1 |++
: 590 0583 1 | FUNCTIONAL DESCRIPTION:
: 591 0584 1 |   This routine is called as part of the V4.0 upgrade.  It finds node
: 592 0585 1 |   numbers in the X25 and X26 Server databases, and, if they have an
: 593 0586 1 |   area number of 0, converts them to an area number supplied by the
: 594 0587 1 |   customer during the upgrade.  (Area 0 is for Phase III nodes only).
: 595 0588 1 |
: 596 0589 1 | INPUTS:
: 597 0590 1 |   PARAM_ID      NMA id for node parameter to convert.
: 598 0591 1 |   RECDSC        Descriptor of the database record to scan for
: 599 0592 1 |   param_id.
: 600 0593 1 | --
: 601 0594 1
: 602 0595 2 BEGIN
: 603 0596 2
: 604 0597 2 LOCAL
: 605 0598 2   node_len,
: 606 0599 2   node_addr: REF BBLOCK,
: 607 0600 2   node_addr_changed;
: 608 0601 2
: 609 0602 2 node_addr_changed = false;
: 610 0603 2
: 611 0604 2 | See if there's a node in the X25 or X29 Server Destination database record.
: 612 0605 2
: 613 0606 2 node_addr = 0;
: 614 0607 2 IF nma$searchfld (.recdsc,
: 615 0608 2   .param_id,
: 616 0609 2   node_len,
: 617 0610 2   node_addr) THEN
: 618 0611 3   BEGIN
: 619 0612 3   IF .node_addr [nma$y_area] EQL 0 AND
: 620 0613 3   .node_addr [nma$w_node] NEQ 0 THEN
: 621 0614 4     BEGIN
: 622 0615 4     node_addr [nma$y_area] = .nml$l_area;
: 623 0616 4     node_addr_changed = true;
: 624 0617 3     END;
: 625 0618 2   END;
: 626 0619 2
: 627 0620 2 RETURN .node_addr_changed;
: 628 0621 1 END;      ! End of NML_FIX_NODE_PARAM
```

0004 00000 NML_FIX_NODE_PARAM:

5E	04	C2	00002	.WORD	Save R2	: 0580
	52	D4	00005	SUBL2	#4, SP	: 0602
	7E	D4	00007	CLRL	NODE_ADDR_CHANGED	: 0606
	5E	DD	00009	CLRL	NODE_ADDR	: 0607
	08	AE	9F	PUSHL	SP	
	04	AC	DD	PUSHAB	NODE_LEN	: 0608
	08	AC	DD	PUSHL	PARAM_ID	: 0607
				PUSHL	RECDSC	

NML\$UPGRADE
V04-000

Node File Routines for Network Management
nml_fix_node_param Upgrade V4.0 node param in p

K 10
16-Sep-1984 00:37:14
14-Sep-1984 12:50:21

VAX-11 Bliss-32 V4.0-742
[NML.SRC]NMLUPGRAD.B32;1

Page 22
(8)

```
00000000G 00 04 FB 00014 CALLS #4, NMASSEARCHFLD
1A 50 E9 0001B BLBC R0, 1$
50 6E D0 0001E MOVL NODE_ADDR, R0
FC 8F 01 A0 93 00021 BITB 1(R0), #252
10 12 00026 BNEQ 1$
60 B5 00028 TSTW (R0)
0C 13 0002A BEQL 1$
0A 00000000' 00 F0 0002C INSV NML$L AREA, #10, #6, (R0)
52 01 D0 00035 MOVL #1, NODE_ADDR_CHANGED
50 52 D0 00038 1$: MOVL NODE_ADDR_CHANGED, R0
04 0003B RET
```

```
:
: 0612
:
: 0613
:
: 0615
: 0616
: 0620
: 0621
```

; Routine Size: 60 bytes, Routine Base: \$CODE\$ + 037D

```
: 629 0622 1
: 630 0623 1 END ! End of module
: 631 0624 1
: 632 0625 0 ELUDOM
```

.EXTRN LIB\$SIGNAL

PSECT SUMMARY

Name	Bytes	Attributes
\$OWNS	12 NOVEC, WRT, RD	, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$PLITS	72 NOVEC, NOWRT, RD	, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$CODE\$	953 NOVEC, NOWRT, RD	, EXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)

Library Statistics

File	Total	Symbols Loaded	Percent	Pages Mapped	Processing Time
\$255\$DUA28:[NML.OBJ]NMLLIB.L32;1	341	50	14	27	00:00.1
\$255\$DUA28:[SHRLIB]NMLIBRY.L32;1	887	15	1	47	00:00.2
\$255\$DUA28:[SYSLIB]STARLET.L32;1	9776	53	0	581	00:02.1

COMMAND QUALIFIERS

```
; BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:NMLUPGRAD/OBJ=OBJ$:NMLUPGRAD MSRC$:NMLUPGRAD/UPDATE=(ENH$:NMLUPGRAD)
; Size: 953 code + 84 data bytes
; Run Time: 00:19.5
```

NML\$UPGRADE
V04-000

Node File Routines for Network Management
nml_fix_node_param Upgrade V4.0 node param in p

L 10
16-Sep-1984 00:37:14

VAX-11 Bliss-32 V4.0-742

Page 23

: Elapsed Time: 00:48.8
: Lines/CPU Min: 1919
: Lexemes/CPU-Min: 16670
: Memory Used: 175 pages
: Compilation Complete

0287 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

