

```

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  MMM      MMM  LLL
NNN      NNNNNN  MMM      MMM  LLL
NNN      NNN  MMM      MMM  LLL
NNN      NNN  MMM      MMM  LLL
NNN      NNN  MMM      MMM  LLL
NNN      NNN  MMM      MMM  LLLLLLLLLLLLLLLLLL
NNN      NNN  MMM      MMM  LLLLLLLLLLLLLLLLLL
NNN      NNN  MMM      MMM  LLLLLLLLLLLLLLLLLL

```

_S

Ps

--

NP

NP

SG

SOI

NP

PA

-L

```

NN      NN  MM      MM  LL      RRRRRRRR  EEEEEEEEE  AAAAAA  LL      000000  GGGGGGGG
NN      NN  MM      MM  LL      RRRRRRRR  EEEEEEEEE  AAAAAA  LL      000000  GGGGGGGG
NN      NN  MMMM    MMMM LL      RR      RR  EE      AA      AA  LL      00      00  GG
NN      NN  MMMM    MMMM LL      RR      RR  EE      AA      AA  LL      00      00  GG
NNNN    NN  MM      MM  LL      RR      RR  EE      AA      AA  LL      00      00  GG
NNNN    NN  MM      MM  LL      RR      RR  EE      AA      AA  LL      00      00  GG
NN      NN  NN  MM      MM  LL      RRRRRRRR  EEEEEEEEE  AA      AA  LL      00      00  GG
NN      NN  NN  MM      MM  LL      RRRRRRRR  EEEEEEEEE  AA      AA  LL      00      00  GG
NN      NN  NN  MM      MM  LL      RR      RR  EE      AAAAAAAAAA LL      00      00  GG  GGGGGG
NN      NN  NN  MM      MM  LL      RR      RR  EE      AAAAAAAAAA LL      00      00  GG  GGGGGG
NN      NN  NN  MM      MM  LL      RR      RR  EE      AA      AA  LL      00      00  GG      GG
NN      NN  NN  MM      MM  LL      RR      RR  EE      AA      AA  LL      00      00  GG      GG
NN      NN  NN  MM      MM  LL      RR      RR  EEEEEEEEE  AA      AA  LLLLLLLLLL 000000  GGGGGG
NN      NN  NN  MM      MM  LLLLLLLLLL RR      RR  EEEEEEEEE  AA      AA  LLLLLLLLLL 000000  GGGGGG

```

```

LL      IIIIII  SSSSSSSS
LL      IIIIII  SSSSSSSS
LL      II     SS
LL      II     SS
LL      II     SS
LL      II     SS
LL      II     SSSSSS
LL      II     SSSSSS
LL      II     SS
LL      II     SS
LL      II     SS
LL      II     SS
LLLLLLLLLLLL IIIIII  SSSSSSSS
LLLLLLLLLLLL IIIIII  SSSSSSSS

```



```

....
....
....
....

```

```

1 0001 0 %TITLE 'NML Read logging parameter module'
2 0002 0 MODULE NML$REALOG (
3 0003 0     LANGUAGE (BLISS32),
4 0004 0     ADDRESSING_MODE (NONEXTERNAL=GENERAL),
5 0005 0     ADDRESSING_MODE (EXTERNAL=GENERAL),
6 0006 0     IDENT = 'V04-000'
7 0007 0 ) =
8 0008 1 BEGIN
9 0009 1
10 0010 1 *****
11 0011 1 *
12 0012 1 *   COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
13 0013 1 *   DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
14 0014 1 *   ALL RIGHTS RESERVED.
15 0015 1 *
16 0016 1 *   THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
17 0017 1 *   ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
18 0018 1 *   INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
19 0019 1 *   COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
20 0020 1 *   OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
21 0021 1 *   TRANSFERRED.
22 0022 1 *
23 0023 1 *   THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
24 0024 1 *   AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
25 0025 1 *   CORPORATION.
26 0026 1 *
27 0027 1 *   DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
28 0028 1 *   SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
29 0029 1 *
30 0030 1 *****
31 0031 1 *****
32 0032 1
33 0033 1
34 0034 1 **
35 0035 1 FACILITY: DECnet-VAX V2.0 Network Management Listener
36 0036 1
37 0037 1 ABSTRACT:
38 0038 1
39 0039 1     This module contains routines for processing the NCP SHOW and LIST
40 0040 1     LOGGING commands.
41 0041 1
42 0042 1 ENVIRONMENT: VAX/VMS Operating System
43 0043 1
44 0044 1 AUTHOR: Distributed Systems Software Engineering
45 0045 1
46 0046 1 CREATION DATE: 30-DEC-1979
47 0047 1
48 0048 1 MODIFIED BY:
49 0049 1
50 0050 1     V03-007 MKP0013      Kathy Perko      12-April-1984
51 0051 1     Add area 1 fix to nodes displayed in logging databases.
52 0052 1
53 0053 1     V03-006 MKP0012      Kathy Perko      21-Mar-1984
54 0054 1     Don't open permanent database if it's not a permanent database
55 0055 1     operation.
56 0056 1
57 0057 1     V03-006 MKP0011      Kathy Perko      5-Aug-1983

```

```

: 58      0058 1      Convert node permanent database to multiple ISAM keys to make it
: 59      0059 1      faster.
: 60      0060 1
: 61      0061 1      V03-006 MKP0010      Kathy Perko      23-Nov-1982
: 62      0062 1      Add module as a source for events.
: 63      0063 1
: 64      0064 1      V03-005 MKP0009      Kathy Perko      18-Oct-1982
: 65      0065 1      Leave permanent database files open until LIST command is
: 66      0066 1      finished. This avoids opening and closing files (notably
: 67      0067 1      the node file for LIST KNOWN LOGGING) several times.
: 68      0068 1
: 69      0069 1      V03-004 MKP0008      Kathy Perko      12-Oct-1982
: 70      0070 1      Report logging events even if executor address is zero.
: 71      0071 1      This allows events to come out for nodes with only PSI.
: 72      0072 1
: 73      0073 1      V03-003 MKP0007      Kathy Perko      20-Sept-1982
: 74      0074 1      Redesign the logging database so that, when the executor
: 75      0075 1      node is the sink node, it is stored with a node address of
: 76      0076 1      zero. LIST LOGGING translates this to the real address of
: 77      0077 1      the executor node. Storing the executor node address this
: 78      0078 1      way allows the logging data base to be transportable to other
: 79      0079 1      nodes.
: 80      0080 1
: 81      0081 1      V03-002 MKP0006      Kathy Perko      10-July-1982
: 82      0082 1      Expand NML$GET_ENTITY_IDS to get entity's with qualifiers.
: 83      0083 1
: 84      0084 1      V03-001 MKP0005      Kathy Perko      22-May-1982
: 85      0085 1      Change NETACP QIO interface to double search key and
: 86      0086 1      add X-25 stuff.
: 87      0087 1
: 88      0088 1      V02-004 MKP0004      Kathy Perko      01-Dec-1981
: 89      0089 1      Supply P3 parameter for calls to NML$NETQIO so that
: 90      0090 1      debug logging only dumps pertinent contents of P4 buffer.
: 91      0091 1
: 92      0092 1      V02-003 MKP0003      Kathy Perko      28-Nov-1981
: 93      0093 1      Fix read summary and events for the executor so
: 94      0094 1      that the parameters are returned in numerical order.
: 95      0095 1
: 96      0096 1      V02-002 MKP0002      Kathy Perko      16-Nov-1981
: 97      0097 1      Add circuits to logging source ids.
: 98      0098 1
: 99      0099 1      V02-001 MKP0001      Kathy Perko      24-July-1981
100      0100 1      Change parameters in call to NML$GET_ENTITY_IDS for
101      0101 1      new QIO interface to NETACP.
102      0102 1      --
103      0103 1

```

```

: 105      0104 1 %SBTTL 'Declarations'
: 106      0105 1
: 107      0106 1
: 108      0107 1  TABLE OF CONTENTS:
: 109      0108 1
: 110      0109 1
: 111      0110 1 FORWARD ROUTINE
: 112      0111 1     nml$readknolog      : NOVALUE,
: 113      0112 1     nml$readactlog     : NOVALUE,
: 114      0113 1     nml$readlogging    : NOVALUE,
: 115      0114 1     nml_lisknosnk      : NOVALUE,
: 116      0115 1     nml_shoknosnk     : NOVALUE,
: 117      0116 1     nml_lislogsnk     : NOVALUE,
: 118      0117 1     nml_shologsnk    : NOVALUE,
: 119      0118 1     nml_readexesnk    : NOVALUE,
: 120      0119 1     nml_lisexesnk     : NOVALUE,
: 121      0120 1     nml_shoexesnk     : NOVALUE,
: 122      0121 1     nml_read_exec_sink : NOVALUE,
: 123      0122 1     nml_readsnknod    : NOVALUE,
: 124      0123 1     nml_format_sink_in_NICE: NOVALUE,
: 125      0124 1     nml_readlogsrc    : NOVALUE;
: 126      0125 1
: 127      0126 1
: 128      0127 1  INCLUDE FILES:
: 129      0128 1
: 130      0129 1
: 131      0130 1 LIBRARY 'LIB$:NMLLIB.L32';
: 132      0131 1 LIBRARY 'SHRLIB$:NMALIBRY.L32';
: 133      0132 1 LIBRARY 'SYSS$LIBRARY:STARLET.L32';
: 134      0133 1
: 135      0134 1
: 136      0135 1  EQUATED SYMBOLS:
: 137      0136 1
: 138      0137 1
: 139      0138 1
: 140      0139 1  OWN STORAGE:
: 141      0140 1
: 142      0141 1
: 143      0142 1
: 144      0143 1  Executor sink node address.
: 145      0144 1
: 146      0145 1 OWN
: 147      0146 1     NML$W_EXEADR : WORD;
: 148      0147 1
: 149      0148 1  Entity buffer and descriptor.
: 150      0149 1
: 151      0150 1 OWN
: 152      0151 1     NML$T_ENTITYBUF : BBLOCK [NML$K_ENTBUFLN],
: 153      0152 1     NML$Q_ENTITYDSC : DESCRIPTOR;
: 154      0153 1
: 155      0154 1
: 156      0155 1  EXTERNAL REFERENCES:
: 157      0156 1
: 158      0157 1
: 159      0158 1 $NML_EXTDEF;           ! Define common external data
: 160      0159 1
: 161      0160 1 EXTERNAL

```

```
: 162      0161 1      nml$gb_ncp_version;  
: 163      0162 1  
: 164      0163 1      EXTERNAL ROUTINE  
: 165      0164 1      NML$OPENFILE,  
: 166      0165 1      NML$SEARCHFLD,  
: 167      0166 1      NML$ADDMSGPRM,  
: 168      0167 1      NML$BLD_REPLY,  
: 169      0168 1      NML$BLDP2,  
: 170      0169 1      NML$ERROR 1,  
: 171      0170 1      NML$GETEXEADR,  
: 172      0171 1      NML$GETINFTABS,  
: 173      0172 1      NML$GET_ENTITY_IDS,  
: 174      0173 1      NML$GETNODNAM,  
: 175      0174 1      NML$GETNXTEVT,  
: 176      0175 1      NML$GETNXTSNK,  
: 177      0176 1      NML$GETCOMFILTERS,  
: 178      0177 1      NML$MATCHRECORD,  
: 179      0178 1      NML$NETQIO,  
: 180      0179 1      NML$SEND;  
: 181      0180 1
```

```

: 183 0181 1 %SBTTL 'NMLSREADKNOLOG Read known logging parameters'
: 184 0182 1 GLOBAL ROUTINE NMLSREADKNOLOG (ENT, INF, DUM1, DUM2) : NOVALUE =
: 185 0183 1
: 186 0184 1 !++
: 187 0185 1 FUNCTIONAL DESCRIPTION:
: 188 0186 1
: 189 0187 1 This routine returns permanent data base information for
: 190 0188 1 all logging sinks.
: 191 0189 1
: 192 0190 1 FORMAL PARAMETERS:
: 193 0191 1
: 194 0192 1 ENT Entity type code.
: 195 0193 1 INF Information type code.
: 196 0194 1 DUM1 Not used.
: 197 0195 1 DUM2 Not used.
: 198 0196 1
: 199 0197 1 IMPLICIT INPUTS:
: 200 0198 1
: 201 0199 1 NONE
: 202 0200 1
: 203 0201 1 IMPLICIT OUTPUTS:
: 204 0202 1
: 205 0203 1 NONE
: 206 0204 1
: 207 0205 1 ROUTINE VALUE:
: 208 0206 1 COMPLETION CODES:
: 209 0207 1
: 210 0208 1 NONE
: 211 0209 1
: 212 0210 1 SIDE EFFECTS:
: 213 0211 1
: 214 0212 1 NONE
: 215 0213 1
: 216 0214 1 --
: 217 0215 1
: 218 0216 2 BEGIN
: 219 0217 2
: 220 0218 2
: 221 0219 2 Return data base information for console, file, and monitor sinks.
: 222 0220 2
: 223 0221 2 NMLSREADLOGGING (.ENT, .INF, NMASC_SNK_CON, 0);
: 224 0222 2 NMLSREADLOGGING (.ENT, .INF, NMASC_SNK_FIL, 0);
: 225 0223 2 NMLSREADLOGGING (.ENT, .INF, NMASC_SNK_MON, 0);
: 226 0224 2
: 227 0225 1 END; ! End of NMLSREADKNOLOG

```

.TITLE NMLSREALOG NML Read logging parameter module
.IDENT \V04-000\

.PSECT \$OWNS,NOEXE,2

0000 NML\$W_EXEADR:
.BLKB 2
0002 .BLKB 2
0004 NML\$T_ENTITYBUF:
.BLKB 64

00044 NML\$Q_ENTITYDSC:
.BLKB 8

- .EXTRN NML\$GB_EVTSRCTYP
- .EXTRN NML\$GQ_EVTSRCDSC
- .EXTRN NML\$GW_EVTCLASS
- .EXTRN NML\$GB_EVTMSKTYP
- .EXTRN NML\$GQ_EVTMSKDSC
- .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_PRMSEM
- .EXTRN NML\$AB_RECBUF, NML\$AL_ENTINFTAB
- .EXTRN NML\$AL_PERMINFTAB
- .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_RECBFDC
- .EXTRN NML\$GW_PRMDESCNT
- .EXTRN NML\$GB_NCP_VERSION
- .EXTRN NML\$OPENFICE, NML\$SEARCHFLD
- .EXTRN NML\$ADDMSGPRM, NML\$BLD_REPLY
- .EXTRN NML\$BLDP2, NML\$ERROR 1
- .EXTRN NML\$GETEXEADR, NML\$GETINFTABS
- .EXTRN NML\$GET_ENTITY_IDS
- .EXTRN NML\$GETRODNAM, NML\$GETNXTEVT
- .EXTRN NML\$GETNXTSNK, NML\$GETCOMFILTERS
- .EXTRN NML\$MATCHRECORD
- .EXTRN NML\$NETQIO, NML\$SEND

.PSECT \$CODES, NOWRT, 2

		0004	0000
52	00000000V	00	9E 00002
7E		01	7D 00009

.ENTRY	NML\$READKNOLOG, Save R2
MOVAB	NML\$READLOGGING, R2
MOVQ	#1, -(SP)

:	0182
:	0221

NML\$REALOG
V04-000

NML Read logging parameter module
NML\$READKNOCLOG Read known logging parameters

M 3
16-Sep-1984 00:29:53
14-Sep-1984 12:50:18

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

Page 7
(3)

N
V

7E	04	AC	7D	0000C	MOVQ	ENT, -(SP)	:
62		04	FB	00010	CALLS	#4, NML\$READLOGGING	:
7E		02	7D	00013	MOVQ	#2, -(SP)	: 0222
7E	04	AC	7D	00016	MOVQ	ENT, -(SP)	:
62		04	FB	0001A	CALLS	#4, NML\$READLOGGING	:
7E		03	7D	0001D	MOVQ	#3, -(SP)	: 0223
7E	04	AC	7D	00020	MOVQ	ENT, -(SP)	:
62		04	FB	00024	CALLS	#4, NML\$READLOGGING	:
		04	00027	RET			: 0225

; Routine Size: 40 bytes, Routine Base: \$CODE\$ + 0000

```

229 0226 1 %SBTTL 'NML$READACTLOG Read active logging parameters'
230 0227 1 GLOBAL ROUTINE NML$READACTLOG (ENT, INF, DUM1, DUM2) : NOVALUE =
231 0228 1
232 0229 1 +-
233 0230 1 FUNCTIONAL DESCRIPTION:
234 0231 1
235 0232 1 This routine returns permanent data base information for
236 0233 1 all active logging sinks.
237 0234 1
238 0235 1 FORMAL PARAMETERS:
239 0236 1
240 0237 1 ENT Entity type code.
241 0238 1 INF Information type code.
242 0239 1 DUM1 Not used.
243 0240 1 DUM2 Not used.
244 0241 1
245 0242 1 IMPLICIT INPUTS:
246 0243 1
247 0244 1 NONE
248 0245 1
249 0246 1 IMPLICIT OUTPUTS:
250 0247 1
251 0248 1 NONE
252 0249 1
253 0250 1 ROUTINE VALUE:
254 0251 1 COMPLETION CODES:
255 0252 1
256 0253 1 NONE
257 0254 1
258 0255 1 SIDE EFFECTS:
259 0256 1
260 0257 1 NONE
261 0258 1
262 0259 1 --
263 0260 1
264 0261 2 BEGIN
265 0262 2
266 0263 2 LOCAL
267 0264 2 BUFEND,
268 0265 2 LISDSC : DESCRIPTOR,
269 0266 2 MSGSIZE,
270 0267 2 PTR,
271 0268 2 SNK,
272 0269 2 STATUS,
273 0270 2 STRTFLG;
274 0271 2
275 0272 2 STRTFLG = FALSE;
276 0273 2
277 0274 2 WHILE NML$GET_ENTITY_IDS (NML$C_SINK, NMASC_ENT_ACT, 0, .STRTFLG, LISDSC) DO
278 0275 3 BEGIN
279 0276 3
280 0277 3 STRTFLG = TRUE;
281 0278 3
282 0279 3 PTR = .LISDSC [DSC$A_POINTER];
283 0280 3 BUFEND = .LISDSC [DSC$A_POINTER] + .LISDSC [DSC$W_LENGTH];
284 0281 3
285 0282 3 WHILE .PTR LSSA .BUFEND DO

```

```

: 286      0283  4      BEGIN
: 287      0284  4
: 288      0285  4      SNK = .(.PTR)<0,32>;
: 289      0286  4      PTR = .PTR + 4;
: 290      0287  4
: 291      0288  4      IF .(.PTR)<0,32> NEQU NMAC_STATE_OFF
: 292      0289  4      THEN
: 293      0290  4          NML$READLOGGING (.ENT, .INF, .SNK, 0);
: 294      0291  4
: 295      0292  4      PTR = .PTR + 4;
: 296      0293  4
: 297      0294  3      END;
: 298      0295  2      END;
: 299      0296  2
: 300      0297  1      END;

```

! End of NML\$READACTLOG

		003C	00000	.ENTRY	NML\$READACTLOG, Save R2,R3,R4,R5	: 0227
5E		08	C2 00002	SUBL2	#8, SP	
		53	D4 00005	CLRL	STRFLG	: 0272
	4008	8F	BB 00007 1\$:	PUSHR	#^M<R3,SP>	: 0274
		7E	D4 00008	CLRL	-(SP)	
7E		02	CE 0000D	MNEGL	#2, -(SP)	
		02	DD 00010	PUSHL	#2	
00000000G	00	05	FB 00012	CALLS	#5, NML\$GET_ENTITY_IDS	
	2F	50	E9 00019	BLBC	R0, 4\$	
	53	01	D0 0001C	MOVL	#1, STRFLG	: 0277
	52	04	AE D0 0001F	MOVL	LISDSC+4, PTR	: 0279
	55	04	6E 3C 00023	MOVZWL	LISDSC, BUFEND	: 0280
	55	04	AE C0 00026	ADDL2	LISDSC+4, BUFEND	
	55	52	D1 0002A 2\$:	CMPL	PTR, BUFEND	: 0282
		D8	1E 0002D	BGEQU	1\$	
	54	82	D0 0002F	MOVL	(PTR)+, SNK	: 0285
	01	62	D1 00032	CMPL	(PTR), #1	: 0288
		0F	13 00035	BEQL	3\$	
		7E	D4 00037	CLRL	-(SP)	: 0290
		54	DD 00039	PUSHL	SNK	
	7E	04	AC 7D 0003B	MOVQ	ENT, -(SP)	
00000000V	00	04	FB 0003F	CALLS	#4, NML\$READLOGGING	
	52	04	C0 00046 3\$:	ADPL2	#4, PTR	: 0292
		DF	11 00049	BRB	2\$: 0282
		04	0004B 4\$:	RET		: 0297

; Routine Size: 76 bytes, Routine Base: \$CODE\$ + 0028

```

: 302 0298 1 %SBTTL 'NML$READLOGGING Read logging parameters'
: 303 0299 1 GLOBAL ROUTINE NML$READLOGGING (ENT, INF, SNK, DUM2) : NOVALUE =
: 304 0300 1
: 305 0301 1 |++
: 306 0302 1 | FUNCTIONAL DESCRIPTION:
: 307 0303 1 |
: 308 0304 1 |     Read logging parameters from the permanent or volatile data bases.
: 309 0305 1 |
: 310 0306 1 | FORMAL PARAMETERS:
: 311 0307 1 |
: 312 0308 1 |     ENT     Entity type code.
: 313 0309 1 |     INF     Information type code.
: 314 0310 1 |     SNK     Sink type code.
: 315 0311 1 |     DUM2    Not used.
: 316 0312 1 |
: 317 0313 1 | --
: 318 0314 1 |
: 319 0315 2 BEGIN
: 320 0316 2
: 321 0317 2 MAP
: 322 0318 2     nml$gb_options : BBLOCK [1];
: 323 0319 2
: 324 0320 2 |
: 325 0321 2 | Open the node data base file.
: 326 0322 2 |
: 327 0323 2 IF .nml$gb_options [nma$v_opt_per] THEN
: 328 0324 2     nml$openfile (nma$c_opn_node, nma$c_opn_ac_ro);
: 329 0325 2
: 330 0326 2 SELECTONEU .inf OF
: 331 0327 2     SET
: 332 0328 2     [nml$c_summary,
: 333 0329 2     nml$c_events]:
: 334 0330 2     BEGIN
: 335 0331 3 |
: 336 0332 3 | Check parse flags to see if this is for KNOWN SINKS or a single
: 337 0333 3 | sink node.
: 338 0334 3 |
: 339 0335 3 |     IF .nml$gl_prs_flg [nml$v_prs_knosnk] THEN
: 340 0336 4 |     BEGIN
: 341 0337 4 |     nml_readexesnk (.ent, .inf, .snk);
: 342 0338 4 |     |
: 343 0339 4 |     | Decide if the operation is on the permanent or volatile data
: 344 0340 4 |     | bases.
: 345 0341 4 |     |
: 346 0342 4 |     | IF .nml$gb_options [nma$v_opt_per] THEN
: 347 0343 4 |     |     nml_lisknosnk (.ent, .inf, .snk)
: 348 0344 4 |     | ELSE
: 349 0345 4 |     |     nml_shoknosnk (.ent, .inf, .snk);
: 350 0346 4 |     | END
: 351 0347 3 |     ELSE
: 352 0348 4 |     BEGIN
: 353 0349 4 |     IF .nml$gl_prs_flg [nml$v_prs_exesnk] THEN
: 354 0350 4 |     nml_readexesnk (.ent, .inf, .snk)
: 355 0351 4 |     ELSE
: 356 0352 5 |     BEGIN
: 357 0353 5 |     |
: 358 0354 5 |     | The NICE command is requesting logging information about

```

```

359 0355 5      | a remote sink node. Now, call the appropriate routine to
360 0356 5      | get the information from the permanent or volatile databases.
361 0357 5
362 0358 5      IF .nml$gb_options [nma$y_opt_per] THEN
363 0359 5          nml_lislogsnk (.ent,
364 0360 5              .inf,
365 0361 5              .snk,
366 0362 5              .nml$gw_evtsnkadr)
367 0363 5      ELSE
368 0364 5          nml_shologsnk (.ent,
369 0365 5              .inf,
370 0366 5              .snk,
371 0367 5              .nml$gw_evtsnkadr);
372 0368 5      END;
373 0369 5      END;
374 0370 5      END;
375 0371 5
376 0372 2      [nml$c_status,
377 0373 2      nml$c_characteristics]:
378 0374 2          nml_readexesnk (nml$c_sink, .inf, .snk);
379 0375 2
380 0376 2      [OTHERWISE]:
381 0377 2          nml$error_1 (nma$c_sts_fun);
382 0378 2
383 0379 2      TES;
384 0380 2      |
385 0381 2      | Close the node data base file later, when the whole command has been
386 0382 2      | completed to avoid multiple opening and closing of the same file.
387 0383 2
388 0384 1      END;
! End of NML$READLOGGING

```

			007C 00000		.ENTRY	NML\$READLOGGING, Save R2,R3,R4,R5,R6	: 0299
	56	00000000V	00	9E 00002	MOVAB	NML_READEXESNK, R6	
	55	00000000G	00	9E 00009	MOVAB	NML\$GB_OPTIONS, R5	
			65	95 00010	TSTB	NML\$GB_OPTIONS	: 0323
			09	18 00012	BGEQ	1\$	
			7E	7C 00014	CLRQ	-(SP)	: 0324
	00000000G	00	02	FB 00016	CALLS	#2, NML\$OPENFILE	
		52	08	AC D0 0001D	MOVL	INF, R2	: 0326
			05	13 00021	BEQL	2\$: 0328
		04		52 D1 00023	CMPL	R2, #4	
			67	12 00026	BNEQ	7\$	
		54	0C	AC D0 00028	MOVL	SNK, R4	: 0337
		53	04	AC D0 0002C	MOVL	ENT, R3	
	23 00000000G	00		02 E1 00030	RBC	#2, NML\$GL_PRS_FLGS+1, 4\$: 0335
			14	BB 00038	PUSHR	#*M<R2,R4>	: 0337
			53	DD 0003A	PUSHL	R3	
		66		03 FB 0003C	CALLS	#3, NML_READEXESNK	
			65	95 0003F	TSTB	NML\$GB_OPTIONS	: 0342
			0C	18 00041	BGEQ	3\$	
			14	BB 00043	PUSHR	#*M<R2,R4>	: 0343
			53	DD 00045	PUSHL	R3	
	0000000C /	00		03 FB 00047	CALLS	#3, NML_LISKNOSNK	

		04	0004E		RET			
		14	BB 0004F	3\$:	PUSHR	#^M<R2,R4>		0345
		53	DD 00051		PUSHL	R3		
00000000V	00	03	FB 00053		CALLS	#3, NML_SHOKNOSNK		
		04	0005A		RET			0335
	06	00	E9 0005B	4\$:	BLBC	NML\$GL_PRS_FLGS+1, 5\$		0349
		14	BB 00062		PUSHR	#^M<R2,R4>		0350
		53	DD 00064		PUSHL	R3		
		37	11 00066		BRB	8\$		
	50	00	3C 00068	5\$:	MOVZWL	NML\$GW_EVTSNKADR, R0		0362
		65	95 0006F		TSTB	NML\$GB_OPTIONS		0358
		0E	18 00071		BGEQ	6\$		
		50	DD 00073		PUSHL	R0		0362
		14	BB 00075		PUSHR	#^M<R2,R4>		0360
00000000V	00	53	DD 00077		PUSHL	R3		0359
		04	FB 00079		CALLS	#4, NML_LISLOGSNK		
		04	00080		RET			
		50	DD 00081	6\$:	PUSHL	R0		0367
		14	BB 00083		PUSHR	#^M<R2,R4>		0365
00000000V	00	53	DD 00085		PUSHL	R3		0364
		04	FB 00087		CALLS	#4, NML_SHOLOGSNK		
		04	0008E		RET			0326
		52	D5 0008F	7\$:	TSTL	R2		0372
		10	13 00091		BEQL	9\$		
	02	52	D1 00093		CMPL	R2, #2		
		0B	1A 00096		BGTRU	9\$		
		AC	DD 00098		PUSHL	SNK		0374
		52	DD 0009B		PUSHL	R2		
		02	DD 0009D		PUSHL	#2		
	66	03	FB 0009F	8\$:	CALLS	#3, NML_READEXESNK		
		04	000A2		RET			
	7E	01	CE 000A3	9\$:	MNEGL	#1, -(SP)		0377
00000000G	00	01	FB 000A6		CALLS	#1, NML\$ERROR_1		
		04	000AD		RET			0384

; Routine Size: 174 bytes, Routine Base: \$CODE\$ + 0074

```

390 0385 1 %SBTTL 'NML_LISKNOSNK List known logging sink node parameters'
391 0386 1 ROUTINE NML_LISKNOSNK (ENT, INF, SNK) : NOVALUE =
392 0387 1
393 0388 1 !++
394 0389 1 ! FUNCTIONAL DESCRIPTION:
395 0390 1 !
396 0391 1 !     This routine returns permanent data base information for
397 0392 1 !     all logging sinks.
398 0393 1 !
399 0394 1 ! FORMAL PARAMETERS:
400 0395 1 !
401 0396 1 !     ENT           Entity type code.
402 0397 1 !     INF           Information type code.
403 0398 1 !     SNK           Sink type code.
404 0399 1 !
405 0400 1 ! IMPLICIT INPUTS:
406 0401 1 !
407 0402 1 !     NONE
408 0403 1 !
409 0404 1 ! IMPLICIT OUTPUTS:
410 0405 1 !
411 0406 1 !     NONE
412 0407 1 !
413 0408 1 ! ROUTINE VALUE:
414 0409 1 ! COMPLETION CODES:
415 0410 1 !
416 0411 1 !     NONE
417 0412 1 !
418 0413 1 ! SIDE EFFECTS:
419 0414 1 !
420 0415 1 !     NONE
421 0416 1 !
422 0417 1 ! --
423 0418 1 !
424 0419 1 ! BEGIN
425 0420 2 !
426 0421 2 ! LOCAL
427 0422 2 !     BLKDSC : DESCRIPTOR,           ! Event parameter descriptor
428 0423 2 !     KEY    : WORD,               ! Record key
429 0424 2 !     RECDSC : DESCRIPTOR,         ! Record descriptor
430 0425 2 !     SNKADR : WORD;               ! Sink node address
431 0426 2 !
432 0427 2 !
433 0428 2 ! List parameters for all sink nodes for this sink type.
434 0429 2 !
435 0430 2 ! KEY = 0;
436 0431 2 ! WHILE NML$MATCHRECORD (.NML$AB_ENTITYDATA [.ENT, EIT$B_FILEID],
437 0432 2 !     NML$GQ_REC$FDSC,
438 0433 2 !     KEY,
439 0434 2 !     .NML$AB_ENTITYDATA [.ENT, EIT$W_KEY], 0, 0,
440 0435 2 !     0, 0, 0, ! No qualifier
441 0436 2 !     RECDSC) DO
442 0437 2 !
443 0438 2 !     BEGIN
444 0439 2 ! Find the sink node address.
445 0440 2 !
446 0441 2 !     BLKDSC [DSC$A_POINTER] = 0;

```

```

447 0442 3 IF NMA$SEARCHFLD (RECDSC,
448 0443 3 NMA$C PCLO SIN,
449 0444 3 BLKDSC [DSC$W_LENGTH],
450 0445 3 BLKDSC [DSC$A_POINTER])
451 0446 3 THEN
452 0447 4 BEGIN
453 0448 4
454 0449 4 SNKADR = .(BLKDSC [DSC$A_POINTER])<0,16>;
455 0450 4
456 0451 4 ! Find the event parameter.
457 0452 4
458 0453 4 BLKDSC [DSC$A_POINTER] = 0;
459 0454 4 IF NMA$SEARCHFLD (RECDSC,
460 0455 4 NMA$C PCLO EVE,
461 0456 4 BLKDSC [DSC$W_LENGTH],
462 0457 4 BLKDSC [DSC$A_POINTER])
463 0458 4 THEN
464 0459 5 BEGIN
465 0460 5
466 0461 5 NML_READSNKNOD (.ENT, .SNK, .SNKADR, BLKDSC);
467 0462 5
468 0463 4 END;
469 0464 4 END;
470 0465 3
471 0466 3 KEY = .KEY + 1;
472 0467 3
473 0468 2 END;
474 0469 2
475 0470 1 END; ! End of NML_LISKOSNR

```

```

001C 00000 NML_LISKOSNR:
54 00000000G 00 9E 00002 .WORD Save R2,R3,R4 : 0386
5E 14 C2 00009 MOVAB NMA$SEARCHFLD, R4
6E B4 0000C SUBL2 #20, SP
2C C5 0000E CLRW KEY : 0430
04 04 AE 9F 00013 1$: MULL3 #44, ENT, R2 : 0434
7E 7C 00016 PUSHAB RECDSC : 0431
7E 7C 00018 CLRQ -(SP)
7E D4 0001A CLRQ -(SP)
0000C000G0042 9F 0001C CLRQ -(SP)
7E 9E 3C 00023 PUSHAB NML$AB_ENTITYDATA+3[R2] : 0434
1C AE 9F 00026 MOVZWL @ (SP)+, -(SP)
00000000G 00 9F 00029 PUSHAB KEY : 0431
00000000G0042 9A 0002F PUSHAB NML$GQ_RECBFDSC
00000000G 00 0A FB 00037 MOVZBL NML$AB_ENTITYDATA[R2], -(SP)
47 50 E9 0003E CALLS #10, NML$MATCHRECORD
10 AE D4 00041 BLBC R0, 3$
10 AE 9F 00044 CLRQ BLKDSC+4 : 0441
10 AE 9F 00047 PUSHAB BLKDSC+4 : 0445
7E C8 8F 9A 0004A PUSHAB BLKDSC : 0444
10 AE 9F 0004E MOVZBL #200, -(SP)
64 04 FB 00051 PUSHAB RECDSC : 0442
CALLS #4, NMA$SEARCHFLD : 0444

```


NML\$REALOG
V04-000

NML Read logging parameter module
NML_LISKNO\$NR List known logging sink node par

H 4
16-Sep-1984 00:29:53
14-Sep-1984 12:50:18

VAX-11 Bliss-32 V4.0-742
[NML.SRC]NMLREALOG.832;1

Page 15
(6)

2D		50	E9	00054	BLBC	R0, 2\$		
53	10	BE	B0	00057	MOVW	@BLKDSC+4, SNKADR	...	0449
	10	AE	D'	0005B	CLRL	BLKDSC+4	...	0453
	10	AE	9F	0005E	PUSHAB	BLKDSC+4	...	0457
	10	AE	9F	00061	PUSHAB	BLKDSC	...	0456
7E	C9	BF	9A	00064	MOVZBL	#201, -(SP)	...	
	10	AE	9F	00068	PUSHAB	RECD\$C	...	0454
64		04	FB	0006B	CALLS	#4, NMA\$SEARCHFLD	...	0456
13		50	E9	0006E	BLBC	R0, 2\$...	
	0C	AE	9F	00071	PUSHAB	BLKDSC	...	0461
7E		53	3C	00074	MOVZWL	SNKADR, -(SP)	...	
	0C	AC	DD	00077	PUSHL	SNK	...	
	04	AC	DD	0007A	PUSHL	ENT	...	
00000000V	00	04	FB	0007D	CALLS	#4, NML_READSNKNOD	...	
		6E	B6	00084	INCW	KEY	...	0466
		8B	11	00086	BRB	1\$...	0431
		04	00088	3\$:	RET		...	0470

; Routine Size: 137 bytes, Routine Base: \$CODE\$ + 0122

```

477 0471 1 %SBTTL 'NML_SHOKNOSNK Show known logging sink node parameters'
478 0472 1 ROUTINE NML_SHOKNOSNK (ENT, INF, SNK) : NOVALUE =
479 0473 1
480 0474 1 !++
481 0475 1 FUNCTIONAL DESCRIPTION:
482 0476 1
483 0477 1 This routine returns permanent data base information for
484 0478 1 all logging sinks.
485 0479 1
486 0480 1 FORMAL PARAMETERS:
487 0481 1
488 0482 1 ENT Entity type code.
489 0483 1 INF Information type code.
490 0484 1 SNK Sink type code.
491 0485 1
492 0486 1 IMPLICIT INPUTS:
493 0487 1
494 0488 1 NONE
495 0489 1
496 0490 1 IMPLICIT OUTPUTS:
497 0491 1
498 0492 1 NONE
499 0493 1
500 0494 1 ROUTINE VALUE:
501 0495 1 COMPLETION CODES:
502 0496 1
503 0497 1 NONE
504 0498 1
505 0499 1 SIDE EFFECTS:
506 0500 1
507 0501 1 NONE
508 0502 1
509 0503 1 !--
510 0504 1
511 0505 2 BEGIN
512 0506 2
513 0507 2 LOCAL
514 0508 2 BUFEND,
515 0509 2 LISDSC : DESCRIPTOR,
516 0510 2 MSGSIZE,
517 0511 2 PTR,
518 0512 2 SNKADR : WORD, ! Sink node address
519 0513 2 STATUS,
520 0514 2 STRTFLG;
521 0515 2
522 0516 2 STRTFLG = FALSE;
523 0517 2
524 0518 2 WHILE NML$GET_ENTITY_IDS (.ENT, NMACS_ENT_KNO, 0, .STRTFLG, LISDSC) DO
525 0519 2 BEGIN
526 0520 2
527 0521 2 PTR = .LISDSC [DSC$A_POINTER];
528 0522 2 BUFEND = .LISDSC [DSC$A_POINTER] + .LISDSC [DSC$W_LENGTH];
529 0523 2
530 0524 2 WHILE .PTR LSSA .BUFEND DO
531 0525 2 BEGIN
532 0526 2
533 0527 2 STRTFLG = TRUE;

```

: 534
: 535
: 536
: 537
: 538
: 539
: 540
: 541
: 542
: 543
: 544

0528 4
0529 4
0530 4
0531 4
0532 4
0533 4
0534 4
0535 3
0536 2
0537 2
0538 1

```
SNKADR = .(.PTR)<0,16>;
NML_SHOLOGSNK (.ENT, .INF, .SNK, .SNKADR);
PTR = .PTR + 4;
END;
END;
END;
! End of NML_SHOKNOSNK
```

			003C 00000	NML_SHOKNOSNK				
	5E		08	C2 00002	.WORD	Save R2,R3,R4,R5		0472
			53	D4 00005	SUBL2	#8, SP		
		4008	8F	BB 00007	1\$: CLRL	STRFLG		0516
			7E	D4 00008	PUSHR	#^M<R3,SP>		0518
	7E		01	CE 0000D	CLRL	-(SP)		
		04	AC	DD 00010	MNEGL	#1, -(SP)		
00000000G	00		05	FB 00013	PUSHL	ENT		
	2C		50	E9 0001A	CALLS	#5, NML\$GET_ENTITY_IDS		
	52	04	AE	D0 0001D	BLBC	R0, 3\$		0521
	55		6E	3C 00021	MOVL	LISDSC+4, PTR		0522
	55	04	AE	C0 00024	MOVZWL	LISDSC, BUFEND		
	55		52	D1 00028	2\$: ADDL2	LISDSC+4, BUFEND		0524
			DA	1E 0002B	CPL	PTR, BUFEND		
	53		01	D0 0002D	EQU	1\$		
	54		62	B0 00030	MOVL	#1, STRFLG		0527
	7E		54	3C 00033	MOVW	(PTR), SNKADR		0529
	7E	08	AC	7D 00036	MOVZWL	SNKADR, -(SP)		0531
		04	AC	DD 0003A	MOVQ	INF, -(SP)		
00000000V	00		04	FB 0003D	PUSHL	ENT		
	52		04	C0 00044	CALLS	#4, NML_SHOLOGSNK		0533
			DF	11 00047	ADDL2	#4, PTR		0524
			04	00049	3\$: BRB	2\$		0538
					RET			

; Routine Size: 74 bytes, Routine Base: \$CODE\$ + 01AB

```

: 546 0539 1 %SBTTL 'NML_LISLOGSNK List logging sink node parameters'
: 547 0540 1 ROUTINE NML_LISLOGSNK (ENT, INF, SNK, SNKADR) : NOVALUE =
: 548 0541 1
: 549 0542 1 +-
: 550 0543 1 FUNCTIONAL DESCRIPTION:
: 551 0544 1
: 552 0545 1 This routine returns permanent data base information for
: 553 0546 1 all logging sinks.
: 554 0547 1
: 555 0548 1 FORMAL PARAMETERS:
: 556 0549 1
: 557 0550 1 ENT Entity type code.
: 558 0551 1 INF Information type code.
: 559 0552 1 SNK Sink type code.
: 560 0553 1 SNKADR Sink node address.
: 561 0554 1
: 562 0555 1 IMPLICIT INPUTS:
: 563 0556 1
: 564 0557 1 NONE
: 565 0558 1
: 566 0559 1 IMPLICIT OUTPUTS:
: 567 0560 1
: 568 0561 1 NONE
: 569 0562 1
: 570 0563 1 ROUTINE VALUE:
: 571 0564 1 COMPLETION CODES:
: 572 0565 1
: 573 0566 1 NONE
: 574 0567 1
: 575 0568 1 SIDE EFFECTS:
: 576 0569 1
: 577 0570 1 NONE
: 578 0571 1
: 579 0572 1 --
: 580 0573 1
: 581 0574 2 BEGIN
: 582 0575 2
: 583 0576 2 MAP
: 584 0577 2 SNKADR : WORD;
: 585 0578 2
: 586 0579 2 LOCAL
: 587 0580 2 BLKDSC : DESCRIPTOR, ! Event parameter descriptor
: 588 0581 2 KEY : WORD, ! Record key
: 589 0582 2 RECDSC : DESCRIPTOR; ! Record descriptor
: 590 0583 2
: 591 0584 2 List parameters for the specified sink node.
: 592 0585 2
: 593 0586 2 KEY = 0;
: 594 0587 2 IF NML$MATCHRECORD (.NML$AB_ENTITYDATA [.ENT, EIT$B_FILEID],
: 595 0588 2 NML$GQ_RECBFDSC,
: 596 0589 2 KEY,
: 597 0590 2 .NML$AB_ENTITYDATA [.ENT, EIT$W_KEY], 2, SNKADR,
: 598 0591 2 0, 0, 0, ! No qualifier
: 599 0592 2 RECDSC)
: 600 0593 2 THEN
: 601 0594 2 BEGIN
: 602 0595 2

```

```

: 603      0596      ! Find the event parameter.
: 604      0597      !
: 605      0598      BLKDSC [DSC$A_POINTER] = 0;
: 606      0599      IF NMA$SEARCHFLD (RECDSC,
: 607      0600      NMA$C_PCLO EVE,
: 608      0601      BLKDSC [DSC$W_LENGTH],
: 609      0602      BLKDSC [DSC$A_POINTER])
: 610      0603      THEN
: 611      0604      NML_READSNKNOD (.ENT, .SNK, .SNKADR, BLKDSC);
: 612      0605      END;
: 613      0606      END;
: 614      0607      END;
: 615      0608      ! End of NML_LISLOGSNK

```

0000 00000 NML_LISLOGSNK:						
	5E	14	C2 00002	.WORD	Save nothing	: 0540
		6E	B4 00005	SUBL2	#20, SP	
		04	AE 9F 00007	CLRW	KEY	: 0586
		7E	7C 0000A	PUSHAB	RECDSC	: 0587
		7E	D4 0000C	CLRQ	-(SP)	
		10	AC 9F 0000E	CLRL	-(SP)	
			02 DD 00011	PUSHAB	SNKADR	
50	04	AC	2C C5 00013	PUSHL	#2	: 0590
			00000000G0040 9F 00018	MULL3	#44, ENT, RO	
		7E	9E 3C 0001F	PUSHAB	NML\$AB_ENTITYDATA+3[RO]	
			1C AE 9F 00022	MOVZWL	@(SP)+, -(SP)	: 0587
			00000000G 00 9F 00025	PUSHAB	KEY	
		7E	00000000G0040 9A 0002B	PUSHAB	NML\$GQ_RECBFDSC	: 0587
00000000G	00		0A FB 00033	MOVZBL	NML\$AB_ENTITYDATA[RO], -(SP)	
	2E		50 E9 0003A	CALLS	#10, NML\$MATCHRECORD	
		10	AE D4 0003D	BLBC	RO, 1\$: 0598
		10	AE 9F 00040	CLRL	BLKDSC+4	: 0602
		10	AE 9F 00043	PUSHAB	BLKDSC+4	: 0601
		7E	C9 8F 9A 00046	PUSHAB	BLKDSC	
		10	AE 9F 0004A	MOVZBL	#201, -(SP)	: 0599
00000000G	00		04 FB 0004D	PUSHAB	RECDSC	: 0601
	14		50 E9 00054	CALLS	#4, NMA\$SEARCHFLD	
		0C	AE 9F 00057	BLBC	RO, 1\$: 0604
	7E	10	AC 3C 0005A	PUSHAB	BLKDSC	
		0C	AC DD 0005E	MOVZWL	SNKADR, -(SP)	: 0604
		04	AC DD 00061	PUSHL	SNK	
00000000V	00		04 FB 00064	PUSHL	ENT	: 0608
			04 0006B 1\$:	CALLS	#4, NML_READSNKNOD	
				RET		

; Routine Size: 108 bytes, Routine Base: \$CODE\$ + 01F5

```

: 617 0609 1 %SBTTL 'NML_SHOLOGSNK Show logging sink node parameters'
: 618 0610 1 ROUTINE NML_SHOLOGSNK (ENT, INF, SNK, SNKADR) : NOVALUE =
: 619 0611 1
: 620 0612 1 !++
: 621 0613 1 ! FUNCTIONAL DESCRIPTION:
: 622 0614 1 !
: 623 0615 1 !     This routine returns volatile data base information for
: 624 0616 1 !     all logging sinks.
: 625 0617 1 !
: 626 0618 1 ! FORMAL PARAMETERS:
: 627 0619 1 !
: 628 0620 1 !     ENT           Entity type code.
: 629 0621 1 !     INF           Information type code.
: 630 0622 1 !     SNK           Sink type code.
: 631 0623 1 !     SNKADR        Sink node address.
: 632 0624 1 !
: 633 0625 1 ! IMPLICIT INPUTS:
: 634 0626 1 !
: 635 0627 1 !     NONE
: 636 0628 1 !
: 637 0629 1 ! IMPLICIT OUTPUTS:
: 638 0630 1 !
: 639 0631 1 !     NONE
: 640 0632 1 !
: 641 0633 1 ! ROUTINE VALUE:
: 642 0634 1 ! COMPLETION CODES:
: 643 0635 1 !
: 644 0636 1 !     NONE
: 645 0637 1 !
: 646 0638 1 ! SIDE EFFECTS:
: 647 0639 1 !
: 648 0640 1 !     NONE
: 649 0641 1 !
: 650 0642 1 ! --
: 651 0643 1 !
: 652 0644 2 BEGIN
: 653 0645 2
: 654 0646 2 MAP
: 655 0647 2     SNKADR : WORD;
: 656 0648 2
: 657 0649 2 LOCAL
: 658 0650 2     BLKDSC   : DESCRIPTOR,           ! Event parameter descriptor
: 659 0651 2     DUMDSC   : REF DESCRIPTOR,      ! Dummy descriptor for table
: 660 0652 2     NFBDSK   : REF DESCRIPTOR,
: 661 0653 2     P2BUFFER : VECTOR [NML$K_P2BUFLN, BYTE],
: 662 0654 2     P2DSC    : DESCRIPTOR,
: 663 0655 2     P3,
: 664 0656 2     PTR;
: 665 0657 2
: 666 0658 2 NML$GETINFTABS (.ENT, .INF, NFBDSK, DUMDSC, 0);
: 667 0659 2 P2DSC [DSC$W_LENGTH] = 80;
: 668 0660 2 P2DSC [DSC$A_POINTER] = P2BUFFER;
: 669 0661 2 NML$BLDP2 (0, .SNKADR, -1, 0, P2DSC, P2DSC);
: 670 0662 2
: 671 0663 2 IF NML$NETQIO (.NFBDSK, P2DSC, P3, NML$GQ_QIOBFDSC)
: 672 0664 2 THEN
: 673 0665 3 BEGIN

```

```

: 674 0666 3
: 675 0667 3
: 676 0668 3
: 677 0669 3
: 678 0670 3
: 679 0671 3
: 680 0672 3
: 681 0673 3
: 682 0674 1

```

```

PTR = .NML$GQ_QIOBFDSC [DSC$A POINTER];
BLKDSC [DSC$W_LENGTH] = .(.PTR)<0,16>;
BLKDSC [DSC$A_POINTER] = .PTR + 2;
NML_READSNKNOD (.ENT, .SNK, .SNKADR, BLKDSC);

END;

```

END; ! End of NML_SHOLOGSNK

```

0000 00000 NML_SHOLOGSNK:
      5E FF7C CE 9E 00002 .WORD Save nothing
      7E D4 00007 MOVAB -132(SP), SP
      04 AE 9F 00009 CLRL -(SP)
      0C AE 9F 0000C PUSHAB DUMDSC
      7E 04 AC 7D 0000F PUSHAB NFB DSC
      00000000G 00 05 FB 00013 MOVQ ENT, -(SP)
      0C AE 50 8F 9B 0001A CALLS #5, NML$GETINFTABS
      10 AE 14 AE 9E 0001F MOVZBW #80, P2DSC
      0C AE 9F 00024 MOVAB P2BUFFER, P2DSC+4
      10 AE 9F 00027 PUSHAB P2DSC
      7E D4 0002A CLRL -(SP)
      7E 01 CE 0002C MNEGL #1, -(SP)
      7E 10 AC 3C 0002F MOVZWL SNKADR, -(SP)
      00000000G 00 7E D4 00033 CLRL -(SP)
      00000000G 00 06 FB 00035 CALLS #6, NML$BLDP2
      00000000G 00 00 9F 0003C PUSHAB NML$GQ_QIOBFDSC
      0C AE 9F 00042 PUSHAB P3
      14 AE 9F 00045 PUSHAB P2DSC
      00000000G 00 10 AE DD 00048 PUSHL NFB DSC
      00000000G 00 04 FB 0004B CALLS #4, NML$NETQIO
      24 50 E9 00052 BLBC R0, 1$
      7C AE 00000000G 00 00 D0 00055 MOVL NML$GQ_QIOBFDSC+4, PTR
      FC AD 60 B0 0005C MOVW (PTR), BLKDSC
      02 A0 9E 00060 MOVAB 2(R0), BLKDSC+4
      7C AE 9F 00065 PUSHAB BLKDSC
      7E 10 AC 3C 00068 MOVZWL SNKADR, -(SP)
      0C AC DD 0006C PUSHL SNK
      04 AC DD 0006F PUSHL ENT
      00000000V 00 04 FB 00072 CALLS #4, NML_READSNKNOD
      04 00079 1$: RET

```

; Routine Size: 122 bytes, Routine Base: \$CODE\$ + 0261

```

: 684 0675 1 %SBTTL 'NML_READEXESNK List executor sink node parameters'
: 685 0676 1 ROUTINE NML_READEXESNK (ENT, INF, SNK) : NOVALUE =
: 686 0677 1
: 687 0678 1 !++
: 688 0679 1 ! FUNCTIONAL DESCRIPTION:
: 689 0680 1
: 690 0681 1 ! This routine returns permanent data base information for
: 691 0682 1 ! all logging sinks.
: 692 0683 1
: 693 0684 1 ! FORMAL PARAMETERS:
: 694 0685 1
: 695 0686 1 ! ENT Entity type code.
: 696 0687 1 ! INF Information type code.
: 697 0688 1 ! SNK Sink type code.
: 698 0689 1
: 699 0690 1 ! --
: 700 0691 1
: 701 0692 2 BEGIN
: 702 0693 2
: 703 0694 2 MAP
: 704 0695 2 NML$GB_OPTIONS : BBLOCK [1];
: 705 0696 2
: 706 0697 2 LOCAL
: 707 0698 2 ENTDESC : DESCRIPTOR, ! Entity descriptor
: 708 0699 2 MSGFLG, ! Response message flag
: 709 0700 2 MSGSIZE, ! Message size
: 710 0701 2 STATUS; ! Temporary status
: 711 0702 2
: 712 0703 2 ! Set up the entity descriptor.
: 713 0704 2
: 714 0705 2 ENTDESC [DSC$W_LENGTH] = 1;
: 715 0706 2 ENTDESC [DSC$A_POINTER] = SNK;
: 716 0707 2
: 717 0708 2 ! Set message flags.
: 718 0709 2
: 719 0710 2 MSGFLG = FALSE; ! No response messages
: 720 0711 2 NML$AB_MSGBLOCK [MSB$L_FLAGS] = MSB$M_ENTD_FLD;
: 721 0712 2 NML$AB_MSGBLOCK [MSB$B_CODE] = NMA$C_STS_SUC;
: 722 0713 2 NML$AB_MSGBLOCK [MSB$A_ENTITY] = ENTDESC;
: 723 0714 2
: 724 0715 2 ! Build the message.
: 725 0716 2
: 726 0717 2 NML$BLD_REPLY (NML$AB_MSGBLOCK, MSGSIZE);
: 727 0718 2
: 728 0719 2 ! Decide if the operation is on the permanent or volatile data base.
: 729 0720 2
: 730 0721 2 IF .NML$GB_OPTIONS [NMA$V_OPT_PER]
: 731 0722 2 THEN
: 732 0723 2 STATUS = NML_LISEXESNK (.ENT, .INF, .SNK, MSGSIZE)
: 733 0724 2 ELSE
: 734 0725 2 STATUS = NML_SHOEXESNK (.ENT, .INF, .SNK, MSGSIZE);
: 735 0726 2
: 736 0727 2 IF .STATUS
: 737 0728 2 THEN
: 738 0729 2 MSGFLG = TRUE;
: 739 0730 2
: 740 0731 2 ! Send the message.

```



```

: 741      0732 2 !
: 742      0733 2 !
: 743      0734 2 !
: 744      0735 2 !
: 745      0736 2 !
: 746      0737 1

```

```

IF .MSGFLG
THEN
    NML$SEND (NML$AB_SNDBUFFER, .MSGSIZE);
END;
! End of NML_READEXESNK

```

				000C 00000 NML_READEXESNK:				
						.WORD	Save R2,R3	: 0676
		53	00000000G	00	9E	00002	MOVAB	NML\$AB_MSGBLOCK, R3
		5E		0C	C2	00009	SUBL2	#12, SP
04		AE		01	B0	0000C	MOVW	#1, ENTDC
08		AE	0C	AC	9E	00010	MOVAB	SNK, ENTDC+4
				52	D4	00015	CLRL	MSGFLG
		63		10	D0	00017	MOVL	#16, NML\$AB_MSGBLOCK
04		A3		01	90	0001A	MOVB	#1, NML\$AB_MSGBLOCK+4
14		A3	04	AE	9E	0001E	MOVAB	ENTDC, NML\$AB_MSGBLOCK+20
			4008	8F	BB	00023	PUSHR	#*M<R3,SP>
00000000G	00			02	FB	00027	CALLS	#2, NML\$BLD_REPLY
			00000000G	00	95	0002E	TSTB	NML\$GB_OPTIONS
				12	18	00034	BGEQ	1\$
				5E	DD	00036	PUSHL	SP
		7E		AC	7D	00038	MOVQ	INF, -(SP)
				AC	DD	0003C	PUSHL	ENT
00000000V	00			04	FB	0003F	CALLS	#4, NML_LISEXESNK
				10	11	00046	BRB	2\$
				5E	DD	00048	PUSHL	SP
		7E		AC	7D	0004A	MOVQ	INF, -(SP)
				AC	DD	0004E	PUSHL	ENT
00000000V	00			04	FB	00051	CALLS	#4, NML_SHOEXESNK
		03		50	E9	00058	BLBC	STATUS, 3\$
		52		01	D0	0005B	MOVL	#1, MSGFLG
		0F		52	E9	0005E	BLBC	MSGFLG, 4\$
				6E	DD	00061	PUSHL	MSGSIZE
00000000G	00		00000000G	00	9F	00063	PUSHAB	NML\$AB_SNDBUFFER
				02	FB	00069	CALLS	#2, NML\$SEND
				04	00	00070	RET	4\$:

: Routine Size: 113 bytes, Routine Base: \$CODE\$ + 02DB

```

748 0738 1 %SBTTL 'NML_LISEXESNK List executor sink node parameters'
749 0739 1 ROUTINE NML_LISEXESNK (ENT, INF, SNK, MSGSIZE) =
750 0740 1
751 0741 1  +-+
752 0742 1  FUNCTIONAL DESCRIPTION:
753 0743 1
754 0744 1      This routine returns permanent data base information for
755 0745 1      all logging sinks.
756 0746 1
757 0747 1  FORMAL PARAMETERS:
758 0748 1
759 0749 1      ENT      Entity type code.
760 0750 1      INF      Information type code.
761 0751 1      SNK      Sink type code.
762 0752 1      MSGSIZE Address of message byte count (current and result).
763 0753 1
764 0754 1  --
765 0755 1
766 0756 2 BEGIN
767 0757 2
768 0758 2 LOCAL
769 0759 2     FLDADR,
770 0760 2     FLDSIZE,
771 0761 2     MSGFLG,           ! Response message flag
772 0762 2     SRCPTR,         ! Source block pointer
773 0763 2     BLKDSC  : DESCRIPTOR, ! Event parameter descriptor
774 0764 2     KEY      : WORD,       ! Record key
775 0765 2     RECDSC  : DESCRIPTOR, ! Record descriptor
776 0766 2     EXEC_ADDR;
777 0767 2
778 0768 2 MSGFLG = FALSE;           ! No response messages
779 0769 2
780 0770 2     Add executor parameters to the output message.
781 0771 2
782 0772 2 KEY = 0;
783 0773 2 IF NML$MATCHRECORD (.NML$AB_ENTITYDATA [NML$C_SINK, EIT$B_FILEID],
784 0774 2     NML$GQ_REC$FDSC,
785 0775 2     KEY,
786 0776 2     .NML$AB_ENTITYDATA [NML$C_SINK, EIT$W_KEY], 1, SNK,
787 0777 2     0, 0, 0,           ! No qualifier
788 0778 2     RECDSC)
789 0779 2 THEN
790 0780 2     BEGIN
791 0781 3     MSGFLG = TRUE;           ! Set response message flag
792 0782 3     SELECTU .INF OF
793 0783 3     SET
794 0784 3     [NML$C_SUMMARY,
795 0785 3     NML$C_STATUS]:
796 0786 4     BEGIN
797 0787 4
798 0788 4     ! If state parameter is defined then add it to the message.
799 0789 4
800 0790 4     FLDADR = 0;
801 0791 4     IF NML$SEARCHFLD (RECDSC,
802 0792 4     NML$C_PCLO_STA,
803 0793 4     FLDSIZE,
804 0794 4     FLDADR)

```

```

805 0795 4 THEN
806 0796 4 NML$ADDMSGPRM (NML$GQ_SNDBFDSC,
807 0797 4 .MSGSIZE,
808 0798 4 NMASC_PCLO_STA,
809 0799 4 NMASM_PTY_COD OR 1,
810 0800 4 1,
811 0801 4 .FLDADR);
812 0802 3 END;
813 0803 3
814 0804 3 [NML$C_SUMMARY,
815 0805 3 NML$C_CHARACTERISTICS]:
816 0806 4 BEGIN
817 0807 4
818 0808 4 | If sink name parameter is defined then add it to the message.
819 0809 4 |
820 0810 4 | FLDADR = 0;
821 0811 4 | IF NMASSEARCHFLD (RECDSC,
822 0812 4 | NMASC_PCLO_LNA,
823 0813 4 | FLDSIZE,
824 0814 4 | FLDADR)
825 0815 4 THEN
826 0816 4 NML$ADDMSGPRM (NML$GQ_SNDBFDSC,
827 0817 4 .MSGSIZE,
828 0818 4 NMASC_PCLO_LNA,
829 0819 4 NMASM_PTY_ASC,
830 0820 4 .FLDSIZE,
831 0821 4 .FLDADR);
832 0822 3 END;
833 0823 3 TES;
834 0824 2 END;
835 0825 2
836 0826 2 | For SUMMARY and EVENT reports, add the sink node ID.
837 0827 2
838 0828 2 IF .INF EQL NML$C_SUMMARY OR
839 0829 2 .INF EQL NML$C_EVENTS THEN
840 0830 2 NML_READ_EXEC_SINK (.INF, .MSGSIZE);
841 0831 2
842 0832 2
843 0833 2 | The executor address is zero in the permanent data base. This allows
844 0834 2 | the database to be transportable to other nodes but not log
845 0835 2 | events to the old executor.
846 0836 2
847 0837 2 EXEC_ADDR = 0;
848 0838 2 SELECT ONEU .INF OF
849 0839 2 SET
850 0840 2 [NML$C_EVENTS,
851 0841 2 NML$C_SUMMARY]:
852 0842 2 BEGIN
853 0843 2 KEY = 0;
854 0844 2 IF NML$MATCHRECORD (.NML$AB_ENTITYDATA [.ENT, EIT$B_FILEID],
855 0845 2 NML$GQ_REC$FDSC,
856 0846 2 KEY,
857 0847 2 .NML$AB_ENTITYDATA [.ENT, EIT$W_KEY], 2, EXEC_ADDR,
858 0848 2 0, 0, 0, ! No qualifier
859 0849 2 RECD$C)
860 0850 3 THEN
861 0851 4 BEGIN

```

```

: 862 0852 4
: 863 0853 4
: 864 0854 4
: 865 0855 4
: 866 0856 4
: 867 0857 4
: 868 0858 4
: 869 0859 4
: 870 0860 4
: 871 0861 5
: 872 0862 5
: 873 0863 5
: 874 0864 6
: 875 0865 6
: 876 0866 6
: 877 0867 6
: 878 0868 6
: 879 0869 6
: 880 0870 5
: 881 0871 4
: 882 0872 3
: 883 0873 2
: 884 0874 2
: 885 0875 2
: 886 0876 1

      | Find the event parameter.
      BLKDSC [DSC$A_POINTER] = 0;
      IF NMA$SEARCHFLD (RECDSC,
                        NMA$C_PCLO_EVE,
                        BLKDSC [DSC$W_LENGTH],
                        BLKDSC [DSC$A_POINTER])
      THEN
      BEGIN
      SRCPTR = 0;
      WHILE NML$GETNXTSNK (BLKDSC, .SNK, SRCPTR) DO
      BEGIN
      MSGFLG = TRUE;      ! Set response message flag
      | Get each event class.
      NML_READLOGSRC (BLKDSC, .SRCPTR, .MSGSIZE);
      END;
      END;
      END;
      TES;
      RETURN .MSGFLG
      END;
      ! End of NML_LISEXESNK
  
```

```

03FC 00000 NML_LISEXESNK:
59 00000000G 00 9E 00002 .WORD Save R2,R3,R4,R5,R6,R7,R8,R9 : 0739
58 00000000G 00 9E 00009 MOVAB NML$ADDMSGPRM, R9
57 00000000G 00 9E 00010 MOVAB NML$GQ SNDBFDSC, R8
56 00000000G 00 9E 00017 MOVAB NML$MATCHRECORD, R7
55 00000000G 00 9E 0001E MOVAB NML$GQ RECBFDSC, R6
54 00000000G 00 9E 00025 MOVAB NMA$SEARCHFLD, R5
5E          24 C2 0002C SUBL2 NML$AB_ENTITYDATA+91, R4
          53 D4 0002F CLR! #36, SP
          0C AE B4 00031 CLR! MSGFLG : 0768
          14 AE 9F 00034 CLR! KEY : 0772
          7F 7C 00037 PUSHAB RECDSC : 0773
          7E D4 00039 CLR! -(SP)
          0C AC 9F 0003B CLR! -(SP)
          01 DD 0003E PUSHAB SNK
7E          64 3C 00040 PUSHL #1
          28 AE 9F 00043 MOVZWL NML$AB_ENTITYDATA+91, -(SP) : 0776
          56 DD 00046 PUSHAB KEY : 0773
7E          FD A4 9A 00048 PUSHL R6
67          0A FB 0004C MOVZBL NML$AB_ENTITYDATA+88, -(SP)
62          50 E9 0004F CALLS #10, NML$MATCH.RECORD
53          01 D0 00052 BLBC R0, 3$
52          08 AC D0 00055 MOVL #1, MSGFLG : 0781
01          52 D1 00059 MOVL INF, R2 : 0782
          24 1A 0005C CMPL R2, #1 : 0784
          6E D4 0005E BGTRU 1$
          CLRL FLDADR : 0790
  
```

			5E DD 00060	PUSHL SP		0791
	08		AE 9F 00062	PUSHAB FLDSIZE		
			7E D4 00065	CLRL -(SP)		
	20		AE 9F 00067	PUSHAB RECDSC		
65			04 FB 0006A	CALLS #4, NML\$SEARCHFLD		
12			50 E9 0006D	BLBC R0, 1\$		
			6E DD 00070	PUSHL FLDADR		0801
			01 DD 00072	PUSHL #1		0796
7E	81		8F 9A 00074	MOVZBL #129, -(SP)		0799
			7E D4 00078	CLRL -(SP)		0796
	10		AC DD 0007A	PUSHL MSGSIZE		0797
			58 DD 0007D	PUSHL R8		0796
69			06 FB 0007F	CALLS #6, NML\$ADDMSGPRM		
			52 D5 00082 1\$:	TSTL R2		0804
			05 13 00084	BEQL 2\$		
02			52 D1 00086	C MPL R2, #2		
			29 12 00089	BNEQ 3\$		
			6E D4 0008B 2\$:	CLRL FLDADR		0810
			5E DD 0008D	PUSHL SP		0811
	08		AE 9F 0008F	PUSHAB FLDSIZE		
7E	64		8F 9A 00092	MOVZBL #100, -(SP)		
	20		AE 9F 00096	PUSHAB RECDSC		
65			04 FB 00099	CALLS #4, NML\$SEARCHFLD		
15			50 E9 0009C	BLBC R0, 3\$		
			6E DD 0009F	PUSHL FLDADR		0821
	08		AE DD 000A1	PUSHL FLDSIZE		0820
7E	40		8F 9A 000A4	MOVZBL #64, -(SP)		0816
7E	64		8F 9A 000A8	MOVZBL #100, -(SP)		
	10		AC DD 000AC	PUSHL MSGSIZE		0817
			58 DD 000AF	PUSHL R8		0816
69			06 FB 000B1	CALLS #6, NML\$ADDMSGPRM		
52	08		AC D0 000B4 3\$:	MOVL INF, R2		0828
			05 13 000B8	BEQL 4\$		
04			52 D1 000BA	C MPL R2, #4		0829
			0C 12 000BD	BNEQ 5\$		
	10		AC DD 000BF 4\$:	PUSHL MSGSIZE		0830
			52 DD 000C2	PUSHL R2		
00000000V	00		02 FB 000C4	CALLS #2, NML_READ_EXEC_SINK		
	08		AE D4 000CB 5\$:	CLRL EXEC_ADDR		0837
			52 D5 000CE	TSTL R2		0840
			05 13 000D0	BEQL 6\$		
	04		52 D1 000D2	C MPL R2, #4		
			6C 12 000D5	BNEQ 8\$		
	0C		AE B4 000D7 6\$:	CLRW KEY		0843
	14		AE 9F 000DA	PUSHAB RECDSC		0844
			7E 7C 000DD	CLRW -(SP)		
			7E D4 000DF	CLRL -(SP)		
	18		AE 9F 000E1	PUSHAB EXEC_ADDR		
			02 DD 000E4	PUSHL #2		
50	04	AC	2C C5 000E6	MULL3 #44, ENT, R0		0847
	A8	A440	9F 000EB	PUSHAB NML\$AB_ENTITYDATA+3[R0]		
		7E	9E 3C 000EF	MOVZWL @(SP)+, -(SP)		
	28		AE 9F 00CF2	PUSHAB KEY		0844
			56 DD 000F5	PUSHL R6		
	7E	A5	A440 9A 000F7	MOVZBL NML\$AB_ENTITYDATA[R0], -(SP)		
	67		0A FB 000FC	CALLS #10, NML\$MATCHRECORD		
	41		50 E9 000FF	BLBC R0, 8\$		

		20	AE	D4	00102	CLRL	BLKDSC+4	:	0855
		20	AE	9F	00105	PUSHAB	BLKDSC+4	:	0859
		20	AE	9F	00108	PUSHAB	BLKDSC	:	0858
	7E	C9	8F	9A	0010B	MOVZBL	#201, -(SP)	:	
		20	AE	9F	0010F	PUSHAB	RECD\$C	:	0856
	65		04	FB	00112	CALLS	#4, NMA\$SEARCHFLD	:	0858
	2B		50	E9	00115	BLBC	RO, 8\$:	
		10	AE	D4	00118	CLRL	SRCPTR	:	0862
		10	AE	9F	0011B	PUSHAB	SRCPTR	:	0863
		0C	AC	DD	0011E	PUSHL	SNK	:	
		24	AE	9F	00121	PUSHAB	BLKDSC	:	
	00000000G	00	03	FB	00124	CALLS	#3, NML\$GETNXTSNK	:	
		15	50	E9	0012B	BLBC	RO, 8\$:	
		53	01	D0	0012E	MOVL	#1, MSGFLG	:	0865
		10	AC	DD	00131	PUSHL	MSGSIZE	:	0869
		14	AE	DD	00134	PUSHL	SRCPTR	:	
		24	AE	9F	00137	PUSHAB	BLKDSC	:	
	00000000V	00	03	FB	0013A	CALLS	#3, NML_READLOGSRC	:	
			0B	11	00141	BRB	7\$:	0863
		50	53	D0	00143	MOVL	MSGFLG, RO	:	0875
			04	00146	8\$:	RET		:	0876

; Routine Size: 327 bytes, Routine Base: \$CODE\$ + 034C

```

888 0877 1 %SBTTL 'NML_SHOEXESNK Show executor sink node parameters'
889 0878 1 ROUTINE NML_SHOEXESNK (ENT, INF, SNK, MSGSIZE) =
890 0879 1
891 0880 1 !++
892 0881 1 FUNCTIONAL DESCRIPTION:
893 0882 1
894 0883 1 This routine returns permanent data base information for
895 0884 1 all logging sinks.
896 0885 1
897 0886 1 FORMAL PARAMETERS:
898 0887 1
899 0888 1 ENT Entity type code.
900 0889 1 INF Information type code.
901 0890 1 SNK Sink type code.
902 0891 1 MSGSIZE Address message byte count (current and result).
903 0892 1
904 0893 1 --
905 0894 1
906 0895 2 BEGIN
907 0896 2
908 0897 2 LOCAL
909 0898 2 DUMDSC : REF DESCRIPTOR,
910 0899 2 MSGFLG, ! Response message flag
911 0900 2 NFBDESC : REF DESCRIPTOR,
912 0901 2 P2BUFFER : VECTOR [NML$K_P2BUFLN, BYTE],
913 0902 2 P2DSC : DESCRIPTOR,
914 0903 2 P3,
915 0904 2 PTR, ! Parameter buffer pointer
916 0905 2 SRCPTR, ! Source block pointer
917 0906 2 BLKDSC : DESCRIPTOR, ! Event parameter descriptor
918 0907 2 STATUS;
919 0908 2
920 0909 2 MSGFLG = FALSE; ! No response messages
921 0910 2 NML$GETINFTABS (NML$C_SINK, .INF, NFBDESC, DUMDSC, 0);
922 0911 2 P2DSC [DSC$W_LENGTH] = NML$K_P2BUFLN;
923 0912 2 P2DSC [DSC$A_POINTER] = P2BUFFER;
924 0913 2 NML$BLDP2 (0, .SNK, -1, 0, P2DSC, P2DSC);
925 0914 2
926 0915 2 STATUS = NML$NETQIO (.NFBDESC, P2DSC, P3, NML$GQ_QIOBFDSC);
927 0916 2
928 0917 2 IF NOT .STATUS
929 0918 2 THEN
930 0919 2 BEGIN
931 0920 2 IF .STATUS NEQ NML$_STS_CMP
932 0921 2 THEN
933 0922 4 BEGIN
934 0923 4 NML_READ_EXEC_SINK (.INF, .MSGSIZE);
935 0924 4 NML$BLD_REPLY (NML$AB_MSGBLOCK, .MSGSIZE);
936 0925 4 RETURN TRUE;
937 0926 3 END;
938 0927 2 END;
939 0928 2
940 0929 2 IF .STATUS
941 0930 2 THEN
942 0931 2 BEGIN
943 0932 2 MSGFLG = TRUE;
944 0933 2 PTR = .NML$GQ_QIOBFDSC [DSC$A_POINTER];

```

```

: 945 0934 SELECTU .INF OF
: 946 0935 SET
: 947 0936 [NML$C_SUMMARY,
: 948 0937 NML$C_STATUS]:
: 949 0938 IF (.PTR)<0,32> NEQU -1
: 950 0939 THEN
: 951 0940 NML$ADDMSGPRM (NML$GQ_SNDBFDSC,
: 952 0941 .MSGSIZE,
: 953 0942 NMASC_PCLO_STA,
: 954 0943 NMASM_PTY_COD OR 1,
: 955 0944 1,
: 956 0945 .PTR);
: 957 0946
: 958 0947 [ALWAYS]:
: 959 0948 PTR = .PTR + 4;
: 960 0949
: 961 0950 [NML$C_SUMMARY,
: 962 0951 NML$C_CHARACTERISTICS]:
: 963 0952 IF (.PTR)<0,16> NEQU 0
: 964 0953 THEN
: 965 0954 NML$ADDMSGPRM (NML$GQ_SNDBFDSC,
: 966 0955 .MSGSIZE,
: 967 0956 NMASC_PCLO_LNA,
: 968 0957 NMASM_PTY_ASC,
: 969 0958 (.PTR)<0,16>,
: 970 0959 .PTR + 2);
: 971 0960
: 972 0961 TES;
: 973 0962 END;
: 974 0963 : For SUMMARY and EVENT reports, add the sink node ID.
: 975 0964
: 976 0965 IF .INF EQL NML$C_SUMMARY OR
: 977 0966 .INF EQL NML$C_EVENTS THEN
: 978 0967 NML_READ_EXEC_SINK (.INF, .MSGSIZE);
: 979 0968
: 980 0969 : List logging events for all sources for this sink node.
: 981 0970
: 982 0971
: 983 0972 SELECTONEU .INF OF
: 984 0973 SET
: 985 0974 [NML$C_EVENTS, NML$C_SUMMARY]:
: 986 0975 BEGIN
: 987 0976 NML$GETINFTABS (NML$C_LOGGING, .INF, NFBFDC, DUMDC, 0);
: 988 0977 P2DSC [DSC$W_LENGTH] = NML$K_P2BUFLN;
: 989 0978 P2DSC [DSC$A_POINTER] = P2BUFFER;
: 990 0979 NML$BLDP2 (0, .NML$W_EXEADR, -1, 0, P2DSC, P2DSC);
: 991 0980
: 992 0981 STATUS = NML$NETQIO (.NFBFDC, P2DSC, P3, NML$GQ_QIOBFDSC);
: 993 0982
: 994 0983 IF .STATUS THEN
: 995 0984 BEGIN
: 996 0985 PTR = .NML$GQ_QIOBFDSC [DSC$A_POINTER];
: 997 0986 BLKDC [DSC$W_LENGTH] = (.PTR)<0,16>;
: 998 0987 BLKDC [DSC$A_POINTER] = .PTR + 2;
: 999 0988 SRCPTR = 0;
: 1000 0989 WHILE NML$GETNXTSNK (BLKDC, .SNK, SRCPTR) DO
: 1001 0990 BEGIN

```



```

: 1002      0991  5      NML_READLOGSRC (BLKDSC, .SRCPTR, .MSGSIZE);
: 1003      0992  5      MSGFLG = TRUE;
: 1004      0993  4      END;
: 1005      0994  4      END
: 1006      0995  3      ELSE
: 1007      0996  4      BEGIN
: 1008      0997  4      IF .STATUS NEQ NML$_STS_CMP
: 1009      0998  4      THEN
: 1010      0999  5      BEGIN
: 1011      1000  5      NML$BLD_REPLY (NML$AB_MSGBLOCK, .MSGSIZE);
: 1012      1001  5      MSGFLG = TRUE;
: 1013      1002  4      END;
: 1014      1003  3      END;
: 1015      1004  2      END;
: 1016      1005  2      TES;
: 1017      1006  2      RETURN .MSGFLG
: 1018      1007  1      END;
    
```

! End of NML_SHOEXESNK

				OFFC 00000 NML_SHOEXESNK:			
				5B 00000000G 00 9E 00002	MOVAB	Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11	0878
				5A 00000000G 00 9E 00009	MOVAB	NML\$BLD_REPLY, R11	
				59 00000000V 00 9E 00010	MOVAB	NML\$AB_MSGBLOCK, R10	
				58 00000000G 00 9E 00017	MOVAB	NML_READ_EXEC_SINK, R9	
				57 00000000G 00 9E 0001E	MOVAB	NML\$NETQIO, R8	
				56 00000000G 00 9E 00025	MOVAB	NML\$BLDP2, R7	
				5E FF78 CE 9E 0002C	MOVAB	NML\$GQ_QIOBFDSC, R6	
					CLRL	-(3(SP), SP	
					CLRL	MSGFLG	0909
					CLRL	-(SP)	0910
				04 AE 9F 00035	PUSHAB	DUMDSC	
				0C AE 9F 00038	PUSHAB	NFBDSC	
				52 08 AC DD 0003B	MOVL	INF, R2	
					PUSHL	R2	
					PUSHL	#2	
					CALLS	#5, NML\$GETINFTABS	
				00000000G 00 05 FB 00043	MOVZBW	#104, P2DSC	0911
				18 AE 68 8F 9B 0004A	MOVAB	P2BUFFER, P2DSC+4	0912
				1C AE 20 AE 9E 0004F	PUSHAB	P2DSC	0913
					PUSHAB	P2DSC	
					CLRL	-(SP)	
				7E 0C 01 CE 0005A	MNEGL	#1, -(SP)	
					PUSHL	SNK	
					CLRL	-(SP)	
				67 06 FB 00064	CALLS	#6, NML\$BLDP2	0915
					PUSHL	R6	
					PUSHAB	P3	
					PUSHAB	P2DSC	
					PUSHL	NFBDSC	
				68 04 FB 00072	CALLS	#4, NML\$NETQIO	
				54 50 DD 00075	MOVL	R0, STATUS	
				20 54 E8 00078	BLBS	STATUS, 2\$	0917
				FFFFF0 54 D1 0007B	CPL	STATUS, #-16	0920
					BEQL	1\$	
				14 13 00082			

		10	AC	DD	00084	PUSHL	MSGSIZE		0923
			52	DD	00087	PUSHL	R2		
69			02	FB	00089	CALLS	#2, NML_READ_EXEC_SINK		
		10	AC	DD	0008C	PUSHL	MSGSIZE		0924
			5A	DD	0008F	PUSHL	R10		
6B			02	FB	00091	CALLS	#2, NML\$BLD_REPLY		
50			01	DO	00094	MOVL	#1, R0		0925
					04	00097	RET		
5D			54	E9	00098	1\$:	BLBC	STATUS, 5\$	0929
55			01	DO	0009B	2\$:	MOVL	#1, MSGFLG	0932
53		04	A6	DO	0009E		MOVL	NML\$GQ_QIOBFDSC+4, PTR	0933
01			52	D1	000A2		CMPL	R2, #1	0936
	FFFFFFF	8F	23	1A	000A5		BGTRU	3\$	
			63	D1	000A7		CMPL	(PTR), #-1	0938
			1A	13	000AE		BEQL	3\$	
			53	DD	000B0		PUSHL	PTR	0945
			01	DD	000B2		PUSHL	#1	0940
7E		81	8F	9A	000B4		MOVZBL	#129, -(SP)	0943
			7E	D4	000B8		CLRL	-(SP)	0940
		10	AC	DD	000BA		PUSHL	MSGSIZE	0941
	00000000G	00	00	9F	000BD		PUSHAB	NML\$GQ_SNDBFDSC	0940
		53	06	FB	000C3		CALLS	#6, NML\$ADDMSGPRM	
			04	CO	000CA	3\$:	ADDL2	#4, PTR	0948
			52	D5	000CD		TSTL	R2	0950
			05	13	000CF		BEQL	4\$	
02			52	D1	000D1		CMPL	R2, #2	
			22	12	000D4		BNEQ	5\$	
			63	B5	000D6	4\$:	TSTW	(PTR)	0952
			1E	13	000D8		BEQL	5\$	
		02	A3	9F	000DA		PUSHAB	2(PTR)	0959
7E			63	3C	000DD		MOVZWL	(PTR), -(SP)	0958
7E		40	8F	9A	000E0		MOVZBL	#64, -(SP)	0954
7E		64	8F	9A	000E4		MOVZBL	#100, -(SP)	
		10	AC	DD	000E8		PUSHL	MSGSIZE	0955
	00000000G	00	00	9F	000EB		PUSHAB	NML\$GQ_SNDBFDSC	0954
			06	FB	000F1		CALLS	#6, NML\$ADDMSGPRM	
			52	D5	000F8	5\$:	TSTL	R2	0965
			05	13	000FA		BEQL	6\$	
04			52	D1	000FC		CMPL	R2, #4	0966
			08	12	000FF		BNEQ	7\$	
		10	AC	DD	00101	6\$:	PUSHL	MSGSIZE	0967
			52	DD	00104		PUSHL	R2	
69			02	FB	00106		CALLS	#2, NML_READ_EXEC_SINK	
			52	D5	00109	7\$:	TSTL	R2	0974
			08	13	0010B		BEQL	8\$	
04			52	D1	0010D		CMPL	R2, #4	
			03	13	00110		BEQL	8\$	
		0094	31	00112		BRW	11\$		
			7E	D4	00115	8\$:	CLRL	-(SP)	0976
		04	AE	9F	00117		PUSHAB	DUMDSC	
		0C	AE	9F	0011A		PUSHAB	NFBFDSC	
			52	DD	0011D		PUSHL	R2	
			01	DD	0011F		PUSHL	#1	
	00000000G	00	05	FB	00121		CALLS	#5, NML\$GETINF TABS	
	18	AE	68	8F	9B	00128	MOVZBW	#104, P2DSC	0977
	1C	AE	20	AE	9E	0012D	MOVAB	P2BUFFER, P2DSC+4	0978
			18	AE	9F	00132	PUSHAB	P2DSC	0979

		1C	AE	9F	00135	PUSHAB	P2DSC		
			7E	D4	00138	CLRL	-(SP)		
	7E		01	CE	0013A	MNEGL	#1, -(SP)		
	7E	00000000'	00	3C	0013D	MOVZWL	NML\$W_EXEADR, -(SP)		
			7E	D4	00144	CLRL	-(SP)		
	67		06	FB	00146	CALLS	#6, NML\$BLDP2		
			56	DD	00149	PUSHL	R6		0981
			0C	AE	9F	0014B	PUSHAB	P3	
			20	AE	9F	0014E	PUSHAB	P2DSC	
			0	AE	DD	00151	PUSHL	NFB DSC	
	68		04	FB	00154	CALLS	#4, NML\$NETQIO		
	54		50	DD	00157	MOVL	R0, STATUS		
	38		54	E9	0015A	BLBC	STATUS, 10\$		0983
	53		04	A6	DD	0015D	MOVL	NML\$GQ_QIOBFDSC+4, PTR	0985
	10	AE	63	B0	00161	MOVW	(PTR), -BLKDSC		0986
	14	AE	02	A3	9E	00165	MOVAB	2(R3), BLKDSC+4	0987
			0C	AE	D4	0016A	CLRL	SRCPTR	0988
			0C	AE	9F	0016D	PUSHAB	SRCPTR	0989
			0C	AC	DD	00170	PUSHL	SNK	
			18	AE	9F	00173	PUSHAB	BLKDSC	
	00000000G	00	03	FB	00176	CALLS	#3, NML\$GETNXTSNK		
		29	50	E9	0017D	BLBC	R0, 11\$		
			10	AC	DD	00180	PUSHL	MSGSIZE	0991
			10	AE	DD	00183	PUSHL	SRCPTR	
			18	AE	9F	00186	PUSHAB	BLKDSC	
	00000000V	00	03	FB	00189	CALLS	#3, NML_READLOGSRC		
		55	01	DD	00190	MOVL	#1, MSGFLG		0992
			D8	11	00193	BRB	9\$		0989
	FFFFFFF0	8F	54	D1	00195	CMPL	STATUS, #-16		0997
			0B	13	0019C	BEQL	11\$		
			10	AC	DD	0019E	PUSHL	MSGSIZE	1000
			5A	DD	001A1	PUSHL	R10		
		6B	02	FB	001A3	CALLS	#2, NML\$BLD_REPLY		
		55	01	DD	001A6	MOVL	#1, MSGFLG		1001
		50	55	DD	001A9	MOVL	MSGFLG, R0		1006
			04	001AC		RET			1007

; Routine Size: 429 bytes, Routine Base: \$CODE\$ + 0493

```

: 1020 1008 1 %SBTTL 'NML_READ_EXEC_SINK Read sink node ID'
: 1021 1009 1 ROUTINE NML_READ_EXEC_SINK (INF, MSGSIZE) : NOVALUE =
: 1022 1010 1
: 1023 1011 1 !++
: 1024 1012 1 FUNCTIONAL DESCRIPTION.
: 1025 1013 1 This routine adds the sink node id to the NICE response
: 1026 1014 1 message.
: 1027 1015 1
: 1028 1016 1 FORMAL PARAMETERS:
: 1029 1017 1 INF Information type code.
: 1030 1018 1 MSGSIZE Address message byte count (current and result).
: 1031 1019 1
: 1032 1020 1 --
: 1033 1021 1
: 1034 1022 2 BEGIN
: 1035 1023 2
: 1036 1024 2 LOCAL
: 1037 1025 2 STATUS; ! Routine completion status
: 1038 1026 2
: 1039 1027 2
: 1040 1028 2 Get executor node address.
: 1041 1029 2
: 1042 1030 2 STATUS = NML$GETEXEADR (NML$W_EXEADR);
: 1043 1031 2
: 1044 1032 2 Add the sink node id to the message if it is required.
: 1045 1033 2
: 1046 1034 2 IF .STATUS THEN
: 1047 1035 3 BEGIN
: 1048 1036 3 SELECTONEU .INF OF
: 1049 1037 3 SET
: 1050 1038 3 [NML$C_EVENTS,
: 1051 1039 3 NML$C_SUMMARY]:
: 1052 1040 4 BEGIN
: 1053 1041 4 nml_format_sink_in_NICE (.nml$w_exeadr, .msgsize);
: 1054 1042 3 END;
: 1055 1043 3 TES;
: 1056 1044 2 END;
: 1057 1045 1 END; ! End of NML_READ_EXEC_SINK

```

```

0004 0000 NML_READ_EXEC_SINK:
: 1009
: 1030
: 1034
: 1036
: 1038
: 1041
: 1045

```

52	00000000	00	DD	00002	WORD	Save R2	:	1009
		52	DD	00009	MOVAB	NML\$W_EXEADR, R2	:	1030
00000000G	00	01	FB	0000B	PUSHL	R2	:	1034
	18	50	E9	00012	CALLS	#1, NML\$GETEXEADR	:	1036
	50	04	AC	D0 00015	BLBC	STATUS, 2\$:	1038
		05	13	00019	MOVL	INF, R0	:	
	04	50	D1	0001B	BEQL	1\$:	
		0D	12	0001E	CML	R0, #4	:	
		08	AC	DD 00020	BNEQ	2\$:	
	7E	62	3C	00023	PUSHL	MSGSIZE	:	1041
00000000V	00	02	FB	00026	MOVZWL	NML\$W_EXEADR, -(SP)	:	
		04	0002D	2\$:	CALLS	#2, NML_FORMAT_SINK_IN_NICE	:	1045
					RET		:	

NML\$REALOG
V04-000

NML Read logging parameter module
NML_READ_EXEC_SINK Read sink node ID

B 6
16-Sep-1984 00:29:53
14-Sep-1984 12:50:18

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

Page 35
(13)

NMI
Tal

; Routine Size: 46 bytes, Routine Base: \$CODE3 + 0640

```

: 1059 1046 1 %SBTTL 'NML_READSNKNOD List logging sink node parameters'
: 1060 1047 1 ROUTINE NML_READSNKNOD (ENT, SNK, SNKADR, BLKDSC) : NOVALUE =
: 1061 1048 1
: 1062 1049 1 |++
: 1063 1050 1 | FUNCTIONAL DESCRIPTION:
: 1064 1051 1 |
: 1065 1052 1 |     Read sink node information from the permanent or volatile data bases.
: 1066 1053 1 |
: 1067 1054 1 | FORMAL PARAMETERS:
: 1068 1055 1 |
: 1069 1056 1 |     ENT           Entity type code.
: 1070 1057 1 |     SNK           Sink type code.
: 1071 1058 1 |     SNKADR        Address of sink node.
: 1072 1059 1 |     BLKDSC        Event parameter descriptor.
: 1073 1060 1 |
: 1074 1061 1 | --
: 1075 1062 1 |
: 1076 1063 2 BEGIN
: 1077 1064 2
: 1078 1065 2 MAP
: 1079 1066 2     NML$GB_OPTIONS : BBLOCK [1],
: 1080 1067 2     SNKADR : WORD,
: 1081 1068 2     BLKDSC : REF DESCRIPTOR;
: 1082 1069 2
: 1083 1070 2 LOCAL
: 1084 1071 2     ENTDESC : DESCRIPTOR,           ! Entity descriptor
: 1085 1072 2     EVTPTTR : REF BBLOCK,       ! Pointer to event block
: 1086 1073 2     MSGFLG,
: 1087 1074 2     MSGSIZE,           ! Message size
: 1088 1075 2     MSKBUF : VECTOR [8, BYTE],    ! Event mask buffer
: 1089 1076 2     SRCPTR : REF BBLOCK;         ! Pointer to source block
: 1090 1077 2
: 1091 1078 2 |
: 1092 1079 2 | If this sink node is the executor node then skip it. Note that the
: 1093 1080 2 | executor node address in the permanent data base is stored as zero.
: 1094 1081 2 | This allows the logging database to be transportable to other nodes.
: 1095 1082 2 |
: 1096 1083 2 IF .NML$GB_OPTIONS [NMA$V_OPT_PER] THEN
: 1097 1084 2     BEGIN
: 1098 1085 2     IF .SNKADR EQL 0 THEN
: 1099 1086 2     RETURN;
: 1100 1087 2     END
: 1101 1088 2 ELSE
: 1102 1089 2     IF .SNKADR EQLU .NML$W_EXEADR THEN
: 1103 1090 2     RETURN;
: 1104 1091 2 |
: 1105 1092 2 | Set up the entity descriptor.
: 1106 1093 2 |
: 1107 1094 2 ENTDESC [DSC$W_LENGTH] = 1;
: 1108 1095 2 ENTDESC [DSC$A_POINTER] = SNK;
: 1109 1096 2 |
: 1110 1097 2 | Set message flags.
: 1111 1098 2 |
: 1112 1099 2 MSGFLG = FALSE;           ! No response messages
: 1113 1100 2 NML$AB_MSGBLOCK [MSB$L_FLAGS] = MSB$M_ENTD_FLD;
: 1114 1101 2 NML$AB_MSGBLOCK [MSB$B_CODE] = NMA$C_STS_SUC;
: 1115 1102 2 NML$AB_MSGBLOCK [MSB$A_ENTITY] = ENTDESC;

```

```

1116 1103 2  Build the message.
1117 1104 2
1118 1105 2
1119 1106 2 NML$BLD_REPLY (NML$AB_MSGBLOCK, MSGSIZE);
1120 1107 2
1121 1108 2  Add sink node id parameter to message.
1122 1109 2
1123 1110 2 nml_format_sink_in_NICE (.snkadr, msgsize);
1124 1111 2
1125 1112 2  List logging for all sources for this sink node.
1126 1113 2
1127 1114 2 SRCPTR = 0;
1128 1115 2 WHILE NML$GETNXTSNK (.BLKDSC, .SNK, SRCPTR) DO
1129 1116 2   BEGIN
1130 1117 2     MSGFLG = TRUE;           ! Set response message flag
1131 1118 2
1132 1119 2     Get each event class.
1133 1120 2
1134 1121 2     NML_READLOGRC (.BLKDSC, .SRCPTR, MSGSIZE);
1135 1122 2     END;
1136 1123 2
1137 1124 2  Send the message.
1138 1125 2
1139 1126 2 IF .MSGFLG THEN
1140 1127 2   NML$SEND (NML$AB_SNDBUFFER, .MSGSIZE);
1141 1128 2
1142 1129 2 RETURN NML$_STS_SUC
1143 1130 1 END;           ! End of NML_READSNKNOD

```

```

001C 00000 NML_READSNKNOD:
54 00000000G 00 9E 00002 .WORD Save R2,R3,R4 : 1047
5E 18 C2 00009 MOVAB NML$AB_MSGBLOCK, R4
52 0C AC 3C 0000C SUBL2 #24, SP : 1085
00000000G 00 95 00010 MOVZWL SNKADR, R2 : 1083
04 18 00016 TSTB NML$GB_OPTIONS
52 05 00018 BGEQ 1$
07 11 0001A TSTL R2 : 1085
52 00000000' 00 B1 0001C 1$: CMPW NML$W_EXEADR, R2 : 1089
10 AE 01 B0 00025 2$: BEQL 5$
14 AE 08 AC 9E 00029 MOVW #1, ENTDC : 1094
53 D4 0002E MOVAB SNK, ENTDC+4 : 1095
04 64 10 D0 00030 CLRL MSGFLG : 1099
04 A4 01 90 00033 MOVL #16, NML$AB_MSGBLOCK : 1100
14 A4 10 AE 9E 00037 MOVB #1, NML$AB_MSGBLOCK+4 : 1101
04 AE 9F 0003C MOVAB ENTDC, NML$AB_MSGBLOCK+20 : 1102
54 DD 0003F PUSHAB MSGSIZE : 1106
00000000G 00 02 FB 00041 PUSHL R4
04 AE 9F 00048 CALLS #2, NML$BLD_REPLY : 1110
52 DD 0004B PUSHAB MSGSIZE
00000000V 00 02 FB 0004D CALLS #2, NML_FORMAT_SINK_IN_NICE
6E D4 00054 CLRL SRCPTR : 1114

```

		08	5E	DD	00056	3\$:	PUSHL	SP		:	1115
		10	AC	DD	00058		PUSHL	SNK		:	
00000000G	00		AC	DD	0005B		PUSHL	BLKDSC		:	
	15		03	FB	0005E		CALLS	#3, NML\$GETNXTSNK		:	
	53		50	E9	00065		BLBC	R0, 4\$:	
			01	DD	00068		MOVL	#1, MSGFLG		:	1117
		04	AE	9F	0006B		PUSHAB	MSGSIZE		:	1121
		04	AE	DD	0006E		PUSHL	SRCPTR		:	
		10	AC	DD	00071		PUSHL	BLKDSC		:	
00000000V	00		03	FB	00074		CALLS	#3, NML_READLOGSRC		:	
	10		D9	11	0007B		BRB	3\$:	1115
			53	E9	0007D	4\$:	BLBC	MSGFLG, 5\$:	1126
		04	AE	DD	00080		PUSHL	MSGSIZE		:	1127
		00000000G	00	9F	00083		PUSHAB	NML\$AB SNDBUFFER		:	
00000000G	00		02	FB	00089		CALLS	#2, NML\$SEND		:	
			04	00090	5\$:		RET			:	1130

; Routine Size: 145 bytes, Routine Base: \$CODE\$ + 066E


```

: 1145 1131 1 %SBTTL 'nml_format_sink_in_NICE Format sink node for NICE response message'
: 1146 1132 1 ROUTINE nml_format_sink_in_NICE (sink_addr, msgsize) : NOVALUE =
: 1147 1133 1
: 1148 1134 1 !++
: 1149 1135 1 ! FUNCTIONAL DESCRIPTION:
: 1150 1136 1 !
: 1151 1137 1 ! Format a sink node from the volatile or permanent logging database
: 1152 1138 1 ! for a NICE response message.
: 1153 1139 1 !
: 1154 1140 1 ! FORMAL PARAMETERS:
: 1155 1141 1 !
: 1156 1142 1 ! SINK_ADDR Node address of sink node.
: 1157 1143 1 ! MSGSIZE Address of current response message size.
: 1158 1144 1 !
: 1159 1145 1 ! --
: 1160 1146 1
: 1161 1147 2 BEGIN
: 1162 1148 2
: 1163 1149 2 MAP
: 1164 1150 2 sink_addr : WORD;
: 1165 1151 2
: 1166 1152 2 LOCAL
: 1167 1153 2 cm_count, ! Coded multiple field count
: 1168 1154 2 prmbuffer : VECTOR [11, BYTE], ! Parameter buffer
: 1169 1155 2 ptr, ! Parameter buffer pointer
: 1170 1156 2 snkbfsc : DESCRIPTOR, ! Sink node name descriptor
: 1171 1157 2 snklen, ! Sink node name length
: 1172 1158 2 snkbuffer : VECTOR [6, BYTE]; ! Buffer for sink node name
: 1173 1159 2
: 1174 1160 2 !
: 1175 1161 2 ! Get sink node name.
: 1176 1162 2 !
: 1177 1163 2 snkbfsc [dsc$w_length] = 6;
: 1178 1164 2 snkbfsc [dsc$a_pointer] = snkbuffer;
: 1179 1165 2 nml$getnodnam (.sink_addr, snkbfsc, snklen);
: 1180 1166 2 !
: 1181 1167 2 ! Add sink node id parameter to message.
: 1182 1168 2 !
: 1183 1169 2 ptr = prmbuffer;
: 1184 1170 2 cm_count = 1;
: 1185 1171 2
: 1186 1172 2 CH$WCHAR_A (2, ptr); ! Move sink node address
: 1187 1173 2 !
: 1188 1174 2 ! If the NCP is Phase III, zero out area numbers in the executor's
: 1189 1175 2 ! area so they make more sense. Node numbers outside the executor's
: 1190 1176 2 ! area will be displayed without formatting the area number and will,
: 1191 1177 2 ! therefore not be very useful, but they will be unique.
: 1192 1178 2 !
: 1193 1179 2 IF CH$RCHAR (nml$gb_ncp_version) LEQ 3 THEN
: 1194 1180 2 BEGIN
: 1195 1181 2 MAP
: 1196 1182 2 nml$w_exadr: BBLOCK [2],
: 1197 1183 2 sink_addr: BBLOCK [2];
: 1198 1184 2
: 1199 1185 2 IF .nml$w_exadr [nma$w_area] EQL .sink_addr [nma$w_area] THEN
: 1200 1186 2 sink_addr [nma$w_area] = 0;
: 1201 1187 2 END;

```

```

: 1202      1188      2 ptr = CH$MOVE (2, sink_addr, .ptr);
: 1203      1189      2 IF .snklen NEQU 0 THEN      ! Move sink node name if present
: 1204      1190      2 BEGIN
: 1205      1191      2   CH$WCHAR_A (nma$m_pty_asc, ptr);
: 1206      1192      2   CH$WCHAR_A (.snklen, ptr);
: 1207      1193      2   ptr = CH$MOVE (.snklen, snkbuffer, .ptr);
: 1208      1194      2   cm_count = 2;
: 1209      1195      2   END;
: 1210      1196      2   !
: 1211      1197      2   ! Add coded multiple sink node id to message.
: 1212      1198      2   !
: 1213      1199      2   nml$addmsgprm (nml$gg_sndbfdsc,
: 1214      1200      2     .msgsize
: 1215      1201      2     nma$c_pc[o_sin,
: 1216      1202      2     nma$m_pty_cmu OR .cm_count,
: 1217      1203      2     .ptr = prmbuffer,
: 1218      1204      2     prmbuffer);
: 1219      1205      1 END;      ! End of format_sink_in_NICE

```

007C 00000 NML_FORMAT_SINK_IN_NICE:						
				.WORD	Save R2,R3,R4,R5,R6	: 1132
				SUBL2	#32, SP	
				MOVW	#6, SNKBFDSC	: 1163
				MOVAB	SNKBUFFER, SNKBFDSC+4	: 1164
				PUSHL	SP	: 1165
				PUSHAB	SNKBFDSC	
				MOVZWL	SINK_ADDR, -(SP)	
				CALLS	#3, NML\$GETNODNAM	
				MOVAB	PRMBUFFER, PTR	: 1169
				MOVL	#1, CM_COUNT	: 1170
				MOVB	#2, (PTR)+	: 1172
				CMPB	NML\$GB_NCP_VERSION, #3	: 1179
				BGTRU	1\$	
				XORB3	NML\$W_EXEADR+1, SINK_ADDR+1, R0	: 1185
				BITB	R0, #252	
				BNEQ	1\$	
				BICB2	#252, SINK_ADDR+1	: 1186
				MOVW	SINK_ADDR, -(PTR)+	: 1188
				MOVL	SNKLEN, R0	: 1189
				BEQL	2\$	
				MOVB	#64, (PTR)+	: 1191
				MOVB	R0, (PTR)+	: 1192
				MOVC3	R0, SNKBUFFER, (PTR)	: 1193
				MOVL	#2, CM_COUNT	: 1194
				PUSHAB	PRMBUFFER	: 1199
				MOVAB	PRMBUFFER, R0	: 1203
				SUBL3	R0, PTR, -(SP)	
				BISL3	#192, CM_COUNT, -(SP)	: 1202
				MOVZBL	#200, -(SP)	: 1199
				PUSHL	MSGSIZE	: 1200
				PUSHAB	NML\$GQ_SNDBFDSC	: 1199
				CALLS	#6, NML\$ADDMSGPRM	
				RET		: 1205

NML\$REALOG
V04-000

NML Read logging parameter module
nml_format_sink_in_NICE Format sink node for

H 6
16-Sep-1984 00:29:53
14-Sep-1984 12:50:18

VAX-11 Bliss-32 V4.0-742
[NML.SRC]NMLREALOG.832;1

Page 41
(15)

NM
VO

; Routine Size: 133 bytes, Routine Base: \$CODES + 06FF

```

: 1221 1206 1 %SBTTL 'NML_READLOGSRC List logging parameters'
: 1222 1207 1 ROUTINE NML_READLOGSRC (DATDSC, SRCPTR, MSGSIZE) : NOVALUE =
: 1223 1208 1
: 1224 1209 1
: 1225 1210 1  !++
: 1226 1211 1  FUNCTIONAL DESCRIPTION:
: 1227 1212 1  Read logging source information from the permanent or volatile data
: 1228 1213 1  bases.
: 1229 1214 1
: 1230 1215 1  FORMAL PARAMETERS:
: 1231 1216 1
: 1232 1217 1  DATDSC      Descriptor of event data.
: 1233 1218 1  SRCPTR      Pointer to event source block.
: 1234 1219 1  MSGSIZE     Current response message size.
: 1235 1220 1
: 1236 1221 1  IMPLICIT INPUTS:
: 1237 1222 1
: 1238 1223 1  NONE
: 1239 1224 1
: 1240 1225 1  IMPLICIT OUTPUTS:
: 1241 1226 1
: 1242 1227 1  NONE
: 1243 1228 1
: 1244 1229 1  ROUTINE VALUE:
: 1245 1230 1  COMPLETION CODES:
: 1246 1231 1
: 1247 1232 1  NONE
: 1248 1233 1
: 1249 1234 1  SIDE EFFECTS:
: 1250 1235 1
: 1251 1236 1  NONE
: 1252 1237 1
: 1253 1238 1  !--
: 1254 1239 1
: 1255 1240 2 BEGIN
: 1256 1241 2
: 1257 1242 2 MAP
: 1258 1243 2  srcptr : REF BBLOCK;
: 1259 1244 2
: 1260 1245 2 LOCAL
: 1261 1246 2  cm_count,      ! Coded multiple field count
: 1262 1247 2  evtptr      : REF BBLOCK,      ! Pointer to event block
: 1263 1248 2  mskbuf      : VECTOR [8, BYTE], ! Event mask buffer
: 1264 1249 2  msklen      : VECTOR [8, BYTE], ! Event mask length
: 1265 1250 2  prmbuffer   : VECTOR [64, BYTE], ! Parameter buffer
: 1266 1251 2  ptr          : VECTOR [64, BYTE], ! Parameter buffer pointer
: 1267 1252 2  snkbfldsc  : DESCRIPTOR,      ! Sink node name descriptor
: 1268 1253 2  snklen      : VECTOR [6, BYTE], ! Sink node name length
: 1269 1254 2  snkbuffer   : VECTOR [6, BYTE], ! Buffer for sink node name
: 1270 1255 2  sink_addr   : BBLOCK [2];    ! Address of sink node
: 1271 1256 2
: 1272 1257 2
: 1273 1258 2  ! Get each event class.
: 1274 1259 2
: 1275 1260 2  evtptr = 0;
: 1276 1261 2  WHILE nml$getnxtvt (.srcptr, evtptr) DO
: 1277 1262 2  BEGIN

```

```

1278      ptr = prmbuffer;
1279      cm_count = 1;
1280
1281      Get source type.
1282
1283      CH$WCHAR_A (nma$m_pty_cod OR 1, ptr);
1284      CH$WCHAR_A (.srcptr [src$b_srctype], ptr);
1285
1286      Get source id.
1287
1288      SELECTONEU .srcptr [src$b_srctype] OF
1289      SET
1290      [nma$c_ent_lin, nma$c_ent_cir, nma$c_ent_mod]:
1291      BEGIN
1292      CH$WCHAR_A (nma$m_pty_asc, ptr);
1293      CH$WCHAR_A (.srcptr [src$b_idlength], ptr);
1294      ptr = CH$MOVE (.srcptr [src$b_idlength],
1295                  srcptr [src$t_id],
1296                  .ptr);
1297
1298      cm_count = .cm_count + 1;
1299      END;
1300      [nma$c_ent_nod]:
1301      BEGIN
1302      Get sink node name.
1303
1304      snkbfsc [dsc$w_length] = 6;
1305      snkbfsc [dsc$a_pointer] = snkbuffer;
1306      nml$getnodnam (.srcptr [src$w_nodadr], snkbfsc, snklen);
1307      CH$WCHAR_A (2, ptr);
1308      sink_addr [nma$w_node] = .srcptr [src$w_nodadr];
1309
1310      If NCP is Phase III, and the node is in the executor's
1311      area, clear the area number. Nodes outside the executor's
1312      area will be BIG numbers.
1313
1314      IF CH$RCHAR (nml$gb_ncp_version) LEQ 3 THEN
1315      BEGIN
1316      MAP
1317      nml$w_exeadr: BBLOCK [2];
1318      IF .sink_addr [nma$v_area] EQL .nml$w_exeadr [nma$v_area] THEN
1319      sink_addr [nma$v_area] = 0;
1320      END;
1321      ptr = CH$MOVE (2, sink_addr [nma$w_node], .ptr);
1322      cm_count = .cm_count + 1;
1323      IF .snklen NEQ 0 THEN      ! Move sink node name if present
1324      BEGIN
1325      CH$WCHAR_A (nma$m_pty_asc, ptr);
1326      CH$WCHAR_A (.snklen, ptr);
1327      ptr = CH$MOVE (.snk(en, snkbuffer, .ptr);
1328      cm_count = .cm_count + 1;
1329      END;
1330      END;
1331      TES;
1332
1333      Get event class.
1334

```

```

: 1335      1320      3      CH$WCHAR_A (2, ptr);
: 1336      1321      3      ptr = CH$MOVE (2, evtptr [evt$w_class], .ptr);
: 1337      1322      3      cm_count = .cm_count + 1;
: 1338      1323      3      |
: 1339      1324      3      |   Get event mask.
: 1340      1325      3      |
: 1341      1326      3      | nml$getcomfilters (.datdsc,
: 1342      1327      3      |                     .evtptr,
: 1343      1328      3      |                     .srcptr [src$b_sinktype],
: 1344      1329      3      |                     mskbuf,
: 1345      1330      3      |                     msklen);
: 1346      1331      3      CH$WCHAR_A (X'20', ptr);
: 1347      1332      3      CH$WCHAR_A (.msklen, ptr);
: 1348      1333      3      ptr = CH$MOVE (.msklen, mskbuf, .ptr);
: 1349      1334      3      cm_count = .cm_count + 1;
: 1350      1335      3      |
: 1351      1336      3      |   Add the parameter to the message.
: 1352      1337      3      |
: 1353      1338      3      | nml$addmsgprm (nml$gg_sndbfdsc,
: 1354      1339      3      |                 .msgsize,
: 1355      1340      3      |                 nma$c_pclo_eve,
: 1356      1341      3      |                 nma$m_pty_cmu OR .cm_count,
: 1357      1342      3      |                 .ptr = prmbuffer,
: 1358      1343      3      |                 prmbuffer);
: 1359      1344      3      |
: 1360      1345      2      | END;
: 1361      1346      1      | END;

```

! End of NML_READLOGSRC

01FC 0000 NML_READLOGSRC:												
										Save R2,R3,R4,R5,R6,R7,R8	: 1207	
	5E	A0	AE	9E	00002					MOVAB	-96(SP), SP	: 1260
			7E	D4	00006					CLRL	EVTPT	: 1261
	56	08	AC	D0	00008					MOVL	SRCPTR, R6	
		4040	8F	BB	0000C	1\$:				PUSHR	#^M<R6,SP>	
00000000G	00		02	FB	00010					CALLS	#2, NML\$GETNXTEVT	
	01		50	E8	00017					BLBS	R0, 2\$	
				04	0001A					RET		
	53	1C	AE	9E	0001B	2\$:				MOVAB	PRMBUFFER, PTR	: 1263
	57		01	D0	0001F					MOVL	#1, CM_COUNT	: 1264
	83	81	8F	90	00022					MOVB	#-127, (PTR)+	: 1268
	83	03	A6	90	00026					MOVB	3(R6), (PTR)+	: 1269
	50	03	A6	9A	0002A					MOVZBL	3(R6), R0	: 1273
	01		50	91	0002E					CMPB	R0, #1	: 1275
			0A	13	00031					BEQL	3\$	
	03		50	91	00033					CMPB	R0, #3	
			18	1F	00036					BLSSU	4\$	
	04		50	91	00038					CMPB	R0, #4	
			13	1A	0003B					BGTRU	4\$	
	83	40	8F	90	0003D	3\$:				MOVB	#64, (PTR)+	: 1277
	83	04	A6	90	00041					MOVB	4(R6), (PTR)+	: 1278
	50	04	A6	9A	00045					MOVZBL	4(R6), R0	: 1279
63	05	A6	50	28	00049					MOVCS	R0, 5(R6), (PTR)	: 1281
			5A	11	0004E					BRB	6\$: 1282

				50	D5	00050	4\$:	TSTL	R0	:	1284	
				58	12	00052		BNEQ	7\$:		
	14	AE		06	B0	00054		MOVW	#6, SNKBF DSC	:	1289	
	18	AE	0C	AE	9E	00058		MOVAB	SNKBUFFER, SNKBF DSC+4	:	1290	
			04	AE	9F	0005D		PUSHAB	SNKLEN	:	1291	
			18	AE	9F	00060		PUSHAB	SNKBF DSC	:		
			04	A6	3C	00063		MOVZWL	4(R6), -(SP)	:		
	00000000G		00	03	FB	00067		CALLS	#3, NML\$GETNODNAM	:		
			83	02	90	0006E		MOVW	#2, (PTR)+	:	1292	
			58	04	A6	B0	00071	MOVW	4(R6), SINK_ADDR	:	1293	
			03	00000000G	00	91	00075	CMPB	NML\$GB_NCP_VERSION, #3	:	1299	
				15	1A	0007C		BGTRU	5\$:		
50	00000000'	00		06	02	EF	0007E	EXTZV	#2, #6, NML\$W_EXEADR+1, R0	:	1303	
50		58		06	0A	ED	00087	CMPZV	#10, #6, SINK_ADDR, R0	:		
				05	12	0008C		BNEQ	5\$:		
				58	FC00	8F	AA	0008E	BICW2	#64512, SINK_ADDR	:	1304
				83		58	B0	00093	MOVW	SINK_ADDR, (PTR)+	:	1306
				57	D6	00096	5\$:	INCL	CM COUNT	:	1307	
				50	04	AE	D0	00098	MOVL	SNRLEN, R0	:	1308
				0E	13	0009C		BEQL	7\$:		
				83	40	8F	90	0009E	MOVW	#64, (PTR)+	:	1310
				83		50	90	000A2	MOVW	R0, (PTR)+	:	1311
	63	0C		AE	50	28	000A5	MOVW	RC, SNKBUFFER, (PTR)	:	1312	
				57	D6	000AA	6\$:	INCL	CM COUNT	:	1313	
				83	02	90	000AC	7\$:	MOVW	#2, (PTR)+	:	1320
				83	00	BE	B0	000AF	MOVW	@EVTPT, (PTR)+	:	1321
				57	D6	000B3		INCL	CM COUNT	:	1322	
				08	AE	9F	000B5	PUSHAB	MSRLEN	:	1326	
				60	AE	9F	000B8	PUSHAB	MSKBUF	:		
				7E	02	A6	9A	000BB	MOVZBL	2(R6), -(SP)	:	1328
				0C	AE	DD	000BF	PUSHL	EVTPT	:	1327	
				04	AC	DD	000C2	PUSHL	DATDSC	:	1326	
	00000000G			00	05	FB	000C5	CALLS	#5, NML\$GETCOMFILTERS	:		
				83	20	90	000CC	MOVW	#32, (PTR)+	:	1331	
				83	08	AE	90	000CF	MOVW	MSKLEN, (PTR)+	:	1332
	63	5C		AE	08	AE	28	000D3	MOVW	MSKLEN, MSKBUF, (PTR)	:	1333
				57	D6	000D9		INCL	CM COUNT	:	1334	
				1C	AE	9F	000DB	PUSHAB	PRMBUFFER	:	1338	
				50	20	AE	9E	000DE	MOVAB	PRMBUFFER, R0	:	1342
	7E			53	50	C3	000E2	SUBL3	R0, PTR, -(SP)	:		
	7E			57	000000C0	8F	C9	000E6	BISL3	#192, CM COUNT, -(SP)	:	1341
				7E		8F	9A	000EE	MOVZBL	#201, -(SP)	:	1338
				0C	AC	DD	000F2	PUSHL	MSGSIZE	:	1339	
	00000000G			00	00000000G	00	9F	000F5	PUSHAB	NML\$GQ SNDBF DSC	:	1338
				06	FB	000FB		CALLS	#6, NML\$ADDMSGPRM	:		
				FF07	31	00102		BRW	1\$:	1261	
				04	00105			RET		:	1346	

; Routine Size: 262 bytes, Routine Base: \$CODE\$ + 0784

NML\$REALOG
V04-000

NML Read logging parameter module
NML_READLOGSRC List logging parameters

M 6
16-Sep-1984 00:29:53
14-Sep-1984 12:50:18

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

Page 46
(17)

NM
VC

: 1363 1347 1 END
: 1364 1348 1
: 1365 1349 0 ELUDOM

! End of module

PSECT SUMMARY

Name	Bytes	Attributes
\$OWNS	76	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$CODES	2186	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	42	12	27	00:00.1
-\$255\$DUA28:[SHRLIB]NMLIBRY.L32;1	887	24	2	47	00:00.2
-\$255\$DUA28:[SYSLIB]STARLET.L32;1	9776	2	0	581	00:02.1

COMMAND QUALIFIERS

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

: Size: 2186 code + 76 data bytes
: Run Time: 00:35.9
: Elapsed Time: 01:15.3
: Lines/CPU Min: 2253
: Lexemes/CPU-Min: 12163
: Memory Used: 182 pages
: Compilation Complete

The image displays a grid of 140 terminal windows, arranged in 10 rows and 14 columns. Each window contains text-based output from a VAX/VMS system. The windows show various types of logs and reports, including:

- NMLREALOG LIS**: A log window showing real-time system activity.
- NMLSEDEST LIS**: A log window showing system destination information.
- NMLSHOPRM LIS**: A log window showing shoproom information.
- NMLSET LIS**: A log window showing system settings.
- DISK**: Windows showing disk status and usage.
- LOG**: Various system logs showing error messages and events.
- REPORTS**: Windows displaying system performance and configuration reports.

The text in the windows is dense and follows a structured format typical of VAX/VMS system output, including headers, data fields, and footers. The overall appearance is that of a multi-terminal session used for system monitoring and management.