

```

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  LLL
NNN      NNN  NNN      MMM  LLL
NNN      NNN  NNN      MMM  LLL
NNN      NNN  NNN      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      AAAAAA  FFFFFFFF  IIIIII  LL      EEEEEEEEE  SSSSSSS
NN      NN      MM      MM      AAAAAA  FFFFFFFF  IIIIII  LL      EEEEEEEEE  SSSSSSS
NN      NN      MMMM   MMMM   AA        AA  FF      II     LL      EE           SS
NN      NN      MMMM   MMMM   AA        AA  FF      II     LL      EE           SS
NNNN    NN      MM      MM      AA        AA  FF      II     LL      EE           SS
NNNN    NN      MM      MM      AA        AA  FF      II     LL      EE           SS
NN  NN   NN      MM      MM      AA        AA  FFFFFFF  II     LL      EEEEEEEEE  SSSSS
NN  NN   NN      MM      MM      AA        AA  FFFFFFF  II     LL      EEEEEEEEE  SSSSS
NN      NNNN   MM      MM      AAAAAAAAA  FF      II     LL      EE           SS
NN      NNNN   MM      MM      AAAAAAAAA  FF      II     LL      EE           SS
NN      NN      MM      MM      AA        AA  FF      II     LL      EE           SS
NN      NN      MM      MM      AA        AA  FF      II     LL      EE           SS
NN      NN      MM      MM      AA        AA  FF      IIIIII  LLLLLLLLLL  EEEEEEEEE  SSSSS
NN      NN      MM      MM      AA        AA  FF      IIIIII  LLLLLLLLLL  EEEEEEEEE  SSSSS
      . . . .
      . . . .
      . . . .
      . . . .

LL      IIIIII  SSSSSSS
LL      IIIIII  SSSSSSS
LL      II     SS
LL      II     SS
LL      II     SS
LL      II     SS
LL      II     SSSSS
LL      II     SSSSS
LL      II     SS
LL      II     SS
LL      II     SS
LL      II     SS
LLLLLLLL  IIIIII  SSSSSSS
LLLLLLLL  IIIIII  SSSSSSS

```

```

1 0001 0 %TITLE 'File Routines for Network Management'
2 0002 0 MODULE NMAFILES (
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 *
13 0013 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
14 0014 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
15 0015 1 * ALL RIGHTS RESERVED.
16 0016 1 *
17 0017 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
18 0018 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
19 0019 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
20 0020 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
21 0021 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
22 0022 1 * TRANSFERRED.
23 0023 1 *
24 0024 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
25 0025 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
26 0026 1 * CORPORATION.
27 0027 1 *
28 0028 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
29 0029 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
30 0030 1 *
31 0031 1 *
32 0032 1 *****
33 0033 1
34 0034 1
35 0035 1 ++
36 0036 1 FACILITY: DECnet Network Management Layer (NMA)
37 0037 1
38 0038 1 ABSTRACT:
39 0039 1
40 0040 1 This module contains routines which manage the files used by
41 0041 1 network management. These files contain permanent data about the
42 0042 1 configuration of the network.
43 0043 1
44 0044 1 ENVIRONMENT: VAX/VMS Operating System
45 0045 1
46 0046 1 AUTHOR: Darrell Duffy , CREATION DATE: 18-December-1979
47 0047 1
48 0048 1 MODIFIED BY:
49 0049 1
50 0050 1 V03-007 MKP0007 Kathy Perko 2-April-1984
51 0051 1 If call is made to open a file and it is already open,
52 0052 1 do a $REWIND to get back to the beginning of the file.
53 0053 1
54 0054 1 V03-006 MKP0006 Kathy Perko 5-Feb-1984
55 0055 1 Fix NMA$READREC so that the correct key is returned to
56 0056 1 the caller.
57 0057 1

```

```
.. 58      0058  1  | V03-005 MKP0005      Kathy Perko      6-Aug-1983
.. 59      0059  1  |      Enhance node permanent database to use multiple ISAM keys
.. 60      0060  1  |      so it's faster to access.  When returning permanent database
.. 61      0061  1  |      records, don't include key in the data returned.
.. 62      0062  1  |
.. 63      0063  1  | V03-004 MKP0004      Kathy Perko      25-April-1983
.. 64      0064  1  |      Allow multiple NMLs to read and update the permanent database
.. 65      0065  1  |      files at once.
.. 66      0066  1  |
.. 67      0067  1  | V03-004 MKP0004      Kathy Perko      25-April-1983
.. 68      0068  1  |      Add NI configurator permanent database.
.. 69      0069  1  |
.. 70      0070  1  | V03-003 MKP0003      Kathy Perko      12-Nov-1982
.. 71      0071  1  |      Allow multiple NMLs to update the permanent database
.. 72      0072  1  |      files at once.
.. 73      0073  1  |
.. 74      0074  1  | V03-002 MKP0002      Kathy Perko      18-Oct-1982
.. 75      0075  1  |      Change the way NML opens and closes files so that it checks
.. 76      0076  1  |      to see if the operation has already been done.  This will
.. 77      0077  1  |      improve the performance of operations which now open and close
.. 78      0078  1  |      various files more than once.
.. 79      0079  1  |
.. 80      0080  1  | V03-001 MKP0001      Kathy Perko      3-Aug-1982
.. 81      0081  1  |      Split module permanent data base into two: one for X25 and
.. 82      0082  1  |      one for X29.
.. 83      0083  1  |
.. 84      0084  1  | V02-001 LMK0001      Len Kawell      27-Jul-1981
.. 85      0085  1  |      Add CIRCUIT and MODULE files.
.. 86      0086  1  |  --
```

```

: 88      0087 1 %SBTTL 'Definitions'
: 89      0088 1
: 90      0089 1
: 91      0090 1 : TABLE OF CONTENTS:
: 92      0091 1
: 93      0092 1
: 94      0093 1 FORWARD ROUTINE
: 95      0094 1     NMA$OPENFILE,      ! Open file by id
: 96      0095 1     NMA$SELECTFILE,    ! Find filedescriptor by fileid
: 97      0096 1     NMA$OPENFAB,      ! Open a file by descriptor
: 98      0097 1     NMA$CLOSEFILE,    ! Close a file by id
: 99      0098 1     NMA$MATCHREC,     ! Find record with specified field
100     0099 1     NMA$READREC,      ! Get a record from a file
101     0100 1     NMA$WRITEREC,     ! Put a record to a file
102     0101 1     NMA$DELETEREC;    ! Delete a record from a file
103     0102 1
104     0103 1
105     0104 1 : INCLUDE FILES:
106     0105 1
107     0106 1
108     0107 1 LIBRARY 'LIBS:NMLLIB.L32';
109     0108 1 LIBRARY 'SHRLIBS:NMALIBRY.L32';
110     0109 1 LIBRARY 'SYSSLIBRARY:STARLET.L32';
111     0110 1
112     0111 1
113     0112 1 : MACROS:
114     0113 1
115     0114 1
116     0115 1
117     0116 1 : Define fields in a file descriptor.
118     0117 1
119     0118 1
120     0119 1 FIELD
121     0120 1     FDSCFLDS =
122     0121 1     SET
123     0122 1         FDSCFNS = [0, 0, 32, 0],
124     0123 1         FDSCFNA = [4, 0, 32, 0],
125     0124 1         FDSCFAB = [8, 0, 32, 0],
126     0125 1         FDSCRAB = [12, 0, 32, 0]
127     0126 1     TES;
128     0127 1
129     0128 1
130     0129 1 : Macro to build file descriptors.
131     0130 1
132     0131 1     FILE          Designator of the file
133     0132 1     FILENAME      Filename string for file
134     0133 1
135     0134 1
136     0135 1 MACRO
137     M 0136 1     $NMA_BLDFILEDSC [FILE, FILENAME] = ! Build as many as you like
138     M 0137 1
139     M 0138 1     OWN          ! Declare the fab and rab
140     M 0139 1         $NAME ('NMA$_', FILE, '_FAB') : $FAB_DECL,
141     M 0140 1         $NAME ('NMA$_', FILE, '_RAB') : $RAB_DECL;
142     M 0141 1
143     M 0142 1     BIND
144     M 0143 1         $NAME ('NMA$_', FILE, '_DSC') =          ! The descriptor

```

```

145 M 0144 1 UPLIT
146 M 0145 1 (
147 M 0146 1 XCHARCOUNT (FILENAME), ! Descriptor of filename str
148 M 0147 1 UPLIT BYTE (FILENAME), ! Addr
149 M 0148 1 XNAME ('NMA$A_', FILE, '_FAB'), ! Fab address
150 M 0149 1 XNAME ('NMA$A_', FILE, '_RAB'), ! Rab address
151 M 0150 1 );
152 M 0151 1 %;
153 M 0152 1 !
154 M 0153 1 !
155 M 0154 1 ! EQUATED SYMBOLS:
156 M 0155 1 !
157 M 0156 1 !
158 M 0157 1 !
159 M 0158 1 ! OWN STORAGE:
160 M 0159 1 !
161 M 0160 1 !
162 M 0161 1 OWN
163 M 0162 1 NMA$W_KEYBUF : WORD; ! Key buffer
164 M 0163 1
165 P 0164 1 $NMA_BLD$FILEDSC
166 P 0165 1 (
167 P 0166 1 NODE, 'NETNODE', ! Remote node database
168 P 0167 1 LINE, 'NETLINE', ! Line database
169 P 0168 1 LOG, 'NETLOGGING', ! Logging database
170 P 0169 1 OBJ, 'NETOBJECT', ! Object database
171 P 0170 1 C:R, 'NETC:R', ! Circuit database
172 P 0171 1 X25, 'NETX25', ! X25 Module database
173 P 0172 1 X29, 'NETX29', ! X29 Module database
174 P 0173 1 CNF, 'NETCONF', ! Ni Configurator Module database
175 M 0174 1 );
176 M 0175 1 !
177 M 0176 1 !
178 M 0177 1 ! EXTERNAL REFERENCES:
179 M 0178 1 !
180 M 0179 1 !
181 M 0180 1 EXTERNAL ROUTINE
182 M 0181 1 NML$DEBUG_MSG,
183 M 0182 1 NML$DEBUG_TXT,
184 M 0183 1 NML$LOGFICEOP,
185 M 0184 1 NML$LOGRECORDOP;
186 M 0185 1

```

```

188 0186 1 %SBTTL 'NMA$OPENFILE Open a specified file'
189 0187 1 GLOBAL ROUTINE NMA$OPENFILE (FILEID, ACCESS) =
190 0188 1
191 0189 1 !++
192 0190 1 ! FUNCTIONAL DESCRIPTION:
193 0191 1
194 0192 1     This routine opens a specified file for specified access.
195 0193 1     The fileid specifies the file, or all files and the access
196 0194 1     specifies read only or read write.
197 0195 1
198 0196 1 ! FORMAL PARAMETERS:
199 0197 1
200 0198 1     FILEID      Value of the fileid parameter (NMA$C_OPN_xxxxx)
201 0199 1     ACCESS      Value of the access parameter (NMA$C_OPN_AC_Rx)
202 0200 1
203 0201 1 ! ROUTINE VALUE:
204 0202 1 ! COMPLETION CODES:
205 0203 1
206 0204 1     Failure or RMS error
207 0205 1
208 0206 1 !--
209 0207 1
210 0208 2 BEGIN
211 0209 2
212 0210 2 LOCAL
213 0211 2     FAB : REF BLOCK [1, BYTE],           ! The fab for the file
214 0212 2     FILEDSC : REF BLOCK [1, BYTE]       ! File descriptor
215 0213 2     FIELD (FDSCFLDS),
216 0214 2     RAB,                                   ! The rab for the file
217 0215 2     STATUS;                               ! Status return
218 0216 2
219 0217 2 IF .FILEID EQL NMA$C_OPN_ALL THEN      ! If ALL
220 0218 3 BEGIN
221 0219 3
222 0220 3     INCRU IDX FROM NMA$C_OPN_MIN         ! Open all the files by
223 0221 3     TO NMA$C_OPN_MAX DO                 ! Calling ourselves
224 0222 4     BEGIN
225 0223 4     STATUS = NMA$OPENFILE (.IDX, .ACCESS); ! Call ourself to open it
226 0224 4     IF NOT .STATUS THEN
227 0225 4     EXITLOOP;
228 0226 4     END
229 0227 3     END
230 0228 2 ELSE
231 0229 3 BEGIN
232 0230 3     STATUS = NMA$ SUCCESS;
233 0231 3     IF NMA$SELECTFILE (.FILEID, FILEDSC) THEN ! Obtain descriptor address
234 0232 4     BEGIN
235 0233 4     FAB = .FILEDSC [FDSCFAB];           ! Get address of FAB
236 0234 4     IF .FAB [FAB$W_IFI] EQL 0 THEN      ! If file isn't open, do it.
237 0235 5     BEGIN
238 0236 5     STATUS = NMA$OPENFAB (.FILEDSC, .ACCESS); ! Open file by descriptor
239 0237 5     IF .STATUS THEN
240 0238 5     NML$LOGFILEOP (DBG$C_FILEIO,
241 0239 5     FILEID,
242 0240 5     $ASCID ('file opened.));
243 0241 5     END
244 0242 4     ELSE

```

```

: 245 0243 4
: 246 0244 4
: 247 0245 4
: 248 0246 4
: 249 0247 5
: 250 0248 5
: 251 0249 5
: 252 0250 4
: 253 0251 4
: 254 0252 3
: 255 0253 3
: 256 0254 2
: 257 0255 2
: 258 0256 2
: 259 0257 1

```

```

      |
      | The file is already open so don't reopen it. However,
      | set RMS's 'next record' back to the beginning of the file.
      |
      | BEGIN
      | RAB = .FILEDSC [FDSCRAB];           ! Point to the rab
      | $REWIND (RAB = .RAB);
      | END;
      |
      | END
ELSE  |
      | RETURN NMA$_BADFID;                 ! If not all, return failure
      | END;
      |
      | RETURN .STATUS
      | END;

```

				.TITLE NMAFILES File Routines for Network Management										
				.IDENT \V04-000\										
				.PSECT \$SPLITS, NOWRT, NOEXE, 2										
	45	44	4F	4E	54	45	4E	00000 P.AAB: .ASCII \NETNODE\	:					
								00007 .BLKB 1	:					
								00008 P.AAA: .LONG 7	:					
	00000000'	00000000'	00000000'	00000000'	00000000'	00000000'	00000	.ADDRESS P.AAB, NMA\$A_NODE_FAB, NMA\$A_NODE_RAB	:					
	45	4E	49	4C	54	45	4E	00018 P.AAD: .ASCII \NETLINE\	:					
								0001F .BLKB 1	:					
								00020 P.AAC: .LONG 7	:					
	00000000'	00000000'	00000000'	00000000'	00000000'	00000000'	00024	.ADDRESS P.AAD, NMA\$A_LINE_FAB, NMA\$A_LINE_RAB	:					
47	4E	49	47	4F	4C	54	45	4E	00030 P.AAF: .ASCII \NETLOGING\	:				
								00039 .BLKB 3	:					
								0003C P.AAE: .LONG 9	:					
	00000000'	00000000'	00000000'	00000000'	00000000'	00000000'	00040	.ADDRESS P.AAF, NMA\$A_LOG_FAB, NMA\$A_LOG_RAB	:					
54	43	45	4A	42	4F	54	45	4E	0004C P.AAH: .ASCII \NETOBJECT\	:				
								00055 .BLKB 3	:					
								00058 P.AAG: .LONG 9	:					
	00000000'	00000000'	00000000'	00000000'	00000000'	00000000'	0005C	.ADDRESS P.AAH, NMA\$A_OBJ_FAB, NMA\$A_OBJ_RAB	:					
	43	52	49	43	54	45	4E	00068 P.AAJ: .ASCII \NETCIRC\	:					
								0006F .BLKB 1	:					
								00070 P.AAI: .LONG 7	:					
	00000000'	00000000'	00000000'	00000000'	00000000'	00000000'	00074	.ADDRESS P.AAJ, NMA\$A_CIR_FAB, NMA\$A_CIR_RAB	:					
	35	32	58	54	45	4E	00080	P.AAL: .ASCII \NETX25\	:					
								00086 .BLKB 2	:					
								00088 P.AAK: .LONG 3	:					
	00000000'	00000000'	00000000'	00000000'	00000000'	00000000'	0008C	.ADDRESS P.AAL, NMA\$A_X25_FAB, NMA\$A_X25_RAB	:					
	39	32	58	54	45	4E	00098	P.AAN: .ASCII \NETX29\	:					
								0009E .BLKB 2	:					
								000A0 P.AAM: .LONG 6	:					
	00000000'	00000000'	00000000'	00000000'	00000000'	00000000'	000A4	.ADDRESS P.AAN, NMA\$A_X29_FAB, NMA\$A_X29_RAB	:					
	46	4E	4F	43	54	45	4E	000B0 P.AAP: .ASCII \NETCONF\	:					
								000B7 .BLKB 1	:					
								000B8 P.AAO: .LONG 7	:					
	00000000'	00000000'	00000000'	00000000'	00000000'	00000000'	000BC	.ADDRESS P.AAP, NMA\$A_CNF_FAB, NMA\$A_CNF_RAB	:					
2E	64	65	6E	65	70	6F	20	65	6C	69	66	000C8	P.AAR: .ASCII \file opened.\	:
												000D4	P.AAQ: .LONG 12	:
												000D8	.ADDRESS P.AAR	:


```

.PSECT $OWNS,NOEXE,2
00000 NMA$W_KEYBUF:
        .BLKB 2
00002        .BLKB 2
00004 NMA$A_NODE_FAB:
        .BLKB 80
00054 NMA$A_NODE_RAB:
        .BLKB 68
00098 NMA$A_LINE_FAB:
        .BLKB 80
000E8 NMA$A_LINE_RAB:
        .BLKB 68
0012C NMA$A_LOG_FAB:
        .BLKB 80
0017C NMA$A_LOG_RAB:
        .BLKB 68
001C0 NMA$A_OBJ_FAB:
        .BLKB 80
00210 NMA$A_OBJ_RAB:
        .BLKB 68
00254 NMA$A_CIR_FAB:
        .BLKB 80
002A4 NMA$A_CIR_RAB:
        .BLKB 68
002E8 NMA$A_X25_FAB:
        .BLKB 80
00338 NMA$A_X25_RAB:
        .BLKB 68
0037C NMA$A_X29_FAB:
        .BLKB 80
003CC NMA$A_X29_RAB:
        .BLKB 68
00410 NMA$A_CNF_FAB:
        .BLKB 80
00460 NMA$A_CNF_RAB:
        .BLKB 68

```

```

NMA$A_NODE_DSC= P.AAA
NMA$A_LINE_DSC= P.AAC
NMA$A_LOG_DSC= P.AAE
NMA$A_OBJ_DSC= P.AAG
NMA$A_CIR_DSC= P.AAI
NMA$A_X25_DSC= P.AAK
NMA$A_X29_DSC= P.AAM
NMA$A_CNF_DSC= P.AAO
.EXTRN NML$DEBUG MSG, NML$DEBUG TXT
.EXTRN NML$LOGFICEOP, NML$LOGRECORDOP
.EXTRN SYSSREWIND

```

.PSECT \$CODE\$,NOWRT,2

```

000007F SE 04 000C 0000
8F AC C2 0002
1A D1 0005
12 0000

```

```

.ENTRY NMA$OPENFILE, Save R2,R3
SUBL2 #4, SP
CMPL FILEID, #127
BNEQ 2$

```

```

: 0187
:
: 0217
:

```

			52	D4	0000F		CLRL	IDX		0223
		08	AC	DD	00011	1\$:	PUSHL	ACCESS		
			52	DD	00014		PUSHL	IDX		
E6	AF		02	FB	00016		CALLS	#2, NMA\$OPENFILE		
	53		50	DO	0001A		MOVL	R0, STATUS		
	5A		53	E9	0001D		BLBC	STATUS, 4\$		0224
			52	D6	00020		INCL	IDX		0220
	07		52	D1	00022		C MPL	IDX, #7		
			EA	1B	00025		BLEQU	1\$		
			51	11	00027		BRB	4\$		0218
	53		01	DO	00029	2\$:	MOVL	#1, STATUS		0230
			5E	DD	0002C		PUSHL	SP		0231
		04	AC	DD	0002E		PUSHL	FILEID		
00000000V	00		02	FB	00031		CALLS	#2, NMA\$SELECTFILE		
	43		50	E9	00038		BLBC	R0, 5\$		
	50		6E	DO	0003B		MOVL	FILEDSC, R0		0233
	51		A0	DO	0003E		MOVL	8(R0), FAB		
		08	A1	B5	00042		TSTW	2(FAB)		0234
			26	12	00045		BNEQ	3\$		
		08	AC	DD	00047		PUSHL	ACCESS		0236
			50	DD	0004A		PUSHL	R0		
00000000V	00		02	FB	0004C		CALLS	#2, NMA\$OPENFAB		
	53		50	DO	00053		MOVL	R0, STATUS		
	21		53	E9	00056		BLBC	STATUS, 4\$		0237
		00000000'	00	9F	00059		PUSHAB	P.AAQ		0240
		04	AC	DD	0005F		PUSHL	FILEID		0239
			01	DD	00062		PUSHL	#1		0238
00000000G	00		03	FB	00064		CALLS	#3, NML\$LOGFILEOP		
			0D	11	0006B		BRB	4\$		0234
	50		A0	DO	0006D	3\$:	MOVL	12(R0), RAB		0248
		0C	50	DD	00071		PUSHL	RAB		0249
00000000G	00		01	FB	00073		CALLS	#1, SYS\$REWIND		
	50		53	DO	0007A	4\$:	MOVL	STATUS, R0		0256
				04	0007D		RET			
			50	D4	0007E	5\$:	CLRL	R0		0257
			04	00080			RET			

: Routine Size: 129 bytes, Routine Base: \$CODE\$ + 0000

```

: 261 0258 1 %SBTTL 'NMA$SELECTFILE Return a file descriptor'
: 262 0259 1 GLOBAL ROUTINE NMA$SELECTFILE (FILEID, FILEDSC) =
: 263 0260 1
: 264 0261 1 +-
: 265 0262 1 FUNCTIONAL DESCRIPTION:
: 266 0263 1
: 267 0264 1 This routine returns the address of the file descriptor for a
: 268 0265 1 specified file. Failure is returned if the fileid is not
: 269 0266 1 valid.
: 270 0267 1
: 271 0268 1 FORMAL PARAMETERS:
: 272 0269 1
: 273 0270 1 FILEID Value of the fileid (NMA$C_OPN_xxxxx)
: 274 0271 1 FILEDSC Address to return address of file descriptor
: 275 0272 1
: 276 0273 1 IMPLICIT INPUTS:
: 277 0274 1
: 278 0275 1 NONE
: 279 0276 1
: 280 0277 1 IMPLICIT OUTPUTS:
: 281 0278 1
: 282 0279 1 NONE
: 283 0280 1
: 284 0281 1 ROUTINE VALUE:
: 285 0282 1 COMPLETION CODES:
: 286 0283 1
: 287 0284 1 Success or failure
: 288 0285 1
: 289 0286 1 SIDE EFFECTS:
: 290 0287 1
: 291 0288 1 NONE
: 292 0289 1
: 293 0290 1 --
: 294 0291 1
: 295 0292 2 BEGIN
: 296 0293 2
: 297 0294 2 LOCAL
: 298 0295 2 STATUS;
: 299 0296 2
: 300 0297 2 STATUS = NMA$_SUCCESS;
: 301 0298 2
: 302 0299 2 .FILEDSC = ! Obtain the file descriptor
: 303 0300 3 BEGIN ! Address
: 304 0301 3
: 305 0302 3 CASE .FILEID FROM NMA$C_OPN_MIN TO NMA$C_OPN_MAX OF
: 306 0303 3 SET
: 307 0304 3
: 308 0305 3 [NMA$C_OPN_NODE]: NMA$A_NODE_DSC;
: 309 0306 3 [NMA$C_OPN_LINE]: NMA$A_LINE_DSC;
: 310 0307 3 [NMA$C_OPN_LOG]: NMA$A_LOG_DSC;
: 311 0308 3 [NMA$C_OPN_OBJ]: NMA$A_OBJ_DSC;
: 312 0309 3 [NMA$C_OPN_CIR]: NMA$A_CIR_DSC;
: 313 0310 3 [NMA$C_OPN_X25]: NMA$A_X25_DSC;
: 314 0311 3 [NMA$C_OPN_X29]: NMA$A_X29_DSC;
: 315 0312 3 [NMA$C_OPN_CNF]: NMA$A_CNF_DSC;
: 316 0313 3 [INRANGE
: 317 0314 3 OUTRANGE]: ! Code not known, fail

```

```

: 318      0315  4      BEGIN
: 319      0316  4
: 320      0317  4      STATUS = NMA$_BADFID;
: 321      0318  4      0      ! Return invalid descriptor
: 322      0319  4
: 323      0320  4      END;
: 324      0321  4
: 325      0322  4      END; TES
: 326      0323  4      END;
: 327      0324  4      RETURN .STATUS
: 328      0325  4
: 329      0326  4
: 330      0327  1      END;

```

0025	003F	07	001F	0038	0019	0031	0014	002B	0004	0000	0000	0002	0009	000C	0011	0019	1\$:	.ENTRY	NMA\$SELECTFILE, Save R2	0259
									00	9E	0000	0002	0009	000C	0011	0019		MOVAB	NMA\$A_NODE_DSC, R2	0297
									01	00	0000	0009	000C	0011	0019			MOVL	#1, STATUS	0302
									AC	CF	0000	000C	0011	0019				CASEL	FILEID, #0, #7	
											2\$-1\$,-							.WORD	2\$-1\$,-	
											3\$-1\$,-								3\$-1\$,-	
											4\$-1\$,-								4\$-1\$,-	
											5\$-1\$,-								5\$-1\$,-	
											6\$-1\$,-								6\$-1\$,-	
											7\$-1\$,-								7\$-1\$,-	
											8\$-1\$,-								8\$-1\$,-	
											9\$-1\$								9\$-1\$	
									50	7C	00021							CLRQ	R0	0315
									30	11	00023							BRB	10\$	0302
					50				62	9E	00025	2\$:						MOVAB	NMA\$A_NODE_DSC, R0	
									2B	11	00028							BRB	10\$	
					50	18			A2	9E	0002A	3\$:						MOVAB	NMA\$A_LINE_DSC, R0	
									25	11	0002E							BRB	10\$	
					50	34			A2	9E	00030	4\$:						MOVAB	NMA\$A_LOG_DSC, R0	
									1F	11	00034							BRB	10\$	
					50	50			A2	9E	00036	5\$:						MOVAB	NMA\$A_OBJ_DSC, R0	
									19	11	0003A							BRB	10\$	
					50	68			A2	9E	0003C	6\$:						MOVAB	NMA\$A_CIR_DSC, R0	
									13	11	00040							BRB	10\$	
					50	0080			C2	9E	00042	7\$:						MOVAB	NMA\$A_X25_DSC, R0	
									0C	11	00047							BRB	10\$	
					50	0098			C2	9E	00049	8\$:						MOVAB	NMA\$A_X29_DSC, R0	
									05	11	0004E							BRB	10\$	
					50	00B0			C2	9E	00050	9\$:						MOVAB	NMA\$A_CNF_DSC, R0	
					08	BC			50	00	00055	10\$:						MOVL	R0, @FILEDSC	0300
									51	00	00059							MOVL	STATUS, R0	0325
									04	00	0005C							RET		0327

; Routine Size: 93 bytes, Routine Base: \$CODE\$ + 0081

```

332 0328 1 %SBTTL 'NMA$OPENFAB Open or Create a File'
333 0329 1 ROUTINE NMA$OPENFAB (FILEDSC, ACCESS) =
334 0330 1
335 0331 1 ++
336 0332 1 FUNCTIONAL DESCRIPTION:
337 0333 1
338 0334 1 This routine does the actual open or create of a file.
339 0335 1 First the fab is loaded with the correct attributes and then
340 0336 1 a create or open service is done. Create is used if the file
341 0337 1 is to be opened with read-write access and the FOP CIF bit is
342 0338 1 specified so that the file is created if it does not exist.
343 0339 1 The created file will be indexed with a two byte binary key.
344 0340 1 A rather large bucket size is used to allow for long records.
345 0341 1 The protection is set to be read for world and group and the
346 0342 1 UIC is set to the system.
347 0343 1
348 0344 1 FORMAL PARAMETERS:
349 0345 1
350 0346 1 FILEDSC Address of the filedescriptor for the file
351 0347 1 ACCESS Value of the access parameter
352 0348 1
353 0349 1 IMPLICIT INPUTS:
354 0350 1
355 0351 1 NONE
356 0352 1
357 0353 1 IMPLICIT OUTPUTS:
358 0354 1
359 0355 1 NONE
360 0356 1
361 0357 1 ROUTINE VALUE:
362 0358 1 COMPLETION CODES:
363 0359 1
364 0360 1 Success or an RMS error
365 0361 1
366 0362 1 SIDE EFFECTS:
367 0363 1
368 0364 1 NONE
369 0365 1
370 0366 1 --
371 0367 1
372 0368 2 BEGIN
373 0369 2
374 0370 2 MAP ! File descriptor format
375 0371 2 FILEDSC : REF BLOCK [1, BYTE] FIELD (FDSCFLDS);
376 0372 2
377 0373 2 LOCAL
378 0374 2 STATUS, ! Return status
379 0375 2 FAB, ! Fab address
380 0376 2 RAB, ! Rab address
381 0377 2 FNS, ! Filename size
382 0378 2 FNA; ! Filename address
383 0379 2
384 0380 2 OWN
385 0381 2 KEYXAB : $XABKEY_DECL, ! Key xab for create
386 0382 2 PROXAB : $XABPRO_DECL; ! Protection xab for create
387 0383 2
388 0384 2 FNA = .FILEDSC [FDSCFNA]; ! Obtain descriptor fields

```

```

389 0385 2 FNS = .FILEDSC [FDSCFNS];
390 0386 2 FAB = .FILEDSC [FDSCFAB];
391 0387 2 RAB = .FILEDSC [FDSCRAB];
392 0388
393 0389 IF .ACCESS EQL NMA$C_OPN_AC_RW ! Check access for read write
394 0390 THEN
395 0391 BEGIN
396 0392
397 P 0393 $FAB_INIT ! Initialize fab for create
398 P 0394 (
399 P 0395 FAB = .FAB, ! Fab address
400 P 0396 BKS = 9, ! Bucket size
401 P 0397 DNM = 'SYS$SYSTEM:.DAT', ! Default filename string
402 P 0398 FAC = (UPD, PUT, GET, DEL), ! File access
403 P 0399 FNA = .FNA, ! Filename string address
404 P 0400 FNS = .FNS, ! Filename string size
405 P 0401 FOP = (CIF, MXV), ! File open codes (create if, max ver)
406 P 0402 ORG = IDX, ! Organization
407 P 0403 RFM = VAR, ! Record format
408 P 0404 SHR = (UPD, PUT, GET, DEL), ! Share
409 P 0405 XAB = PROXAB ! Xab chain
410 0406 );
411 0407
412 P 0408 $XABKEY_INIT ! Initialize key xab
413 P 0409 (
414 P 0410 XAB = KEYXAB, ! Xab address
415 P 0411 DTP = BN2, ! 2 byte binary
416 P 0412 POSO = 0, ! Position
417 P 0413 SIZE = 2, ! Size
418 P 0414 KREF = 0 ! Key reference (primary)
419 0415 );
420 0416
421 P 0417 $XABPRO_INIT ! Initialize protection xab
422 P 0418 (
423 P 0419 XAB = PROXAB, ! Xab address
424 P 0420 UIC = (1, 4), ! Uic of owner (system)
425 P 0421 PRO = (RWED, RWED, , ), ! Protection (group and world no access)
426 P 0422 NXT = KEYXAB ! Chain
427 0423 );
428 0424
429 0425 STATUS = $CREATE (FAB = .FAB); ! Create the file if not found
430 0426
431 0427 END
432 0428
433 0429 ELSE
434 0430
435 0431 BEGIN
436 P 0432 $FAB_INIT ! Initialize the fab
437 P 0433 (
438 P 0434 FAB = .FAB, ! Fab address
439 P 0435 FAC = (GET), ! File access
440 P 0436 FNA = .FNA, ! Filename string address
441 P 0437 FNS = .FNS, ! Filename string size
442 P 0438 DNM = 'SYS$SYSTEM:.DAT', ! Default filename string
443 P 0439 SHR = (UPD, PUT, GET, DEL) ! Share
444 0440 );
445 0441

```

```

: 446      0442      3          STATUS = $OPEN (FAB = .FAB);      ! Open the file
: 447      0443      2
: 448      0444      2          END;
: 449      0445      2
: 450      0446      2          IF NOT .STATUS                ! Return failure status
: 451      0447      2          THEN
: 452      0448      2          RETURN .STATUS;
: 453      0449      2
: 454      P 0450      2          $RAB_INIT                    ! Initialize the rab
: 455      P 0451      2          ?
: 456      P 0452      2          RAB = .RAB,                  ! Rab address
: 457      P 0453      2          FAB = .FAB,                  ! Fab address
: 458      P 0454      2          KBF = NMA$W_KEYBUF,          ! Key buffer address
: 459      P 0455      2          KRF = 0,                      ! Key of reference
: 460      P 0456      2          KSZ = 2,                      ! Key size
: 461      P 0457      2          RAC = KEY,                    ! Record access mode
: 462      P 0458      2          ROP = (UIF,KGE)              ! Record options (put is update)
: 463      0459      2          );
: 464      0460      2
: 465      0461      2          RETURN $CONNECT (RAB = .RAB);  ! Connect record stream and return
: 466      0462      2
: 467      0463      1          END;

```

```

                                .PSECT $SPLITS,NOWRT,NOEXE,2
54  41  44  2E  3A  4D  45  54  53  59  53  24  53  59  53  000DC P.AAS: .ASCII \SYS$SYSTEM:.DAT\
54  41  44  2E  3A  4D  45  54  53  59  53  24  53  59  53  000EB P.AAT: .ASCII \SYS$SYSTEM:.DAT\
                                .PSECT $OWNS,NOEXE,2
                                004A4 KEYXAB: .BLKB 76
                                004F0 PROXAB: .BLKB 88
                                $RMS_PTR= KEYXAB
                                $RMS_PTR= PROXAB
                                .EXTRN SYSS$CREATE, SYSS$OPEN
                                .EXTRN SYSS$CONNECT
                                .PSECT $CODE$,NOWRT,2
                                07FC 0000 NMA$OPENFAB:
                                .WORD Save R2,R3,R4,R5,R6,R7,R8,R9,R10 : 0329
                                5A 00000000' 00 9E 00002 MOVAB PROXAB, R10 : 0384
                                50 04 AC D0 00009 MOVL FILEDSC, R0
                                58 04 A0 D0 0000D MOVL 4(R0), FNA : 0385
                                59 60 D0 00011 MOVL (R0), FNS : 0386
                                56 08 A0 7D 00014 MOVQ 8(R0), FAB : 0389
                                01 08 AC D1 00018 CMPL ACCESS, #1
                                03 13 0001C BEQL 1$
                                0081 31 0001E BRW 2$
                                0050 8F 00 6E 00 2C 00021 1$: MOVCS #0, (SP), #0, #80, (FAB) : 0406
                                66 66 5003 8F 60 00029 MOVW #20483, (FAB)
                                04 A6 02000002 8F D0 0002E MOVL #33554434, 4(FAB)
                                16 A6 0F0F 8F B0 00036 MOVW #3855, 22(FAB)

```

004C	8F	00	1D A6	20 90 0003C	MOVB	#32, 29(FAB)	:	
			1F A6	02 90 0004J	MOVB	#2, 31(FAB)	:	
			24 A6	6A 9E 00044	MOVAB	PROXAB, 36(FAB)	:	
			2C A6	58 D0 00048	MOVL	FNA, 44(FAB)	:	
			30 A6 00000000'	00 9E 0004C	MOVAB	P.AAS, 48(FAB)	:	
			34 A6	59 90 00054	MOVB	FNS, 52(FAB)	:	
			35 A6	0F 90 00058	MOVB	#15, 53(FAB)	:	
			3E A6	09 90 0005C	MOVB	#9, 62(FAB)	:	
			6E	00 2C 00060	MOVCS	#0, (SP), #0, #76, \$RMS_PTR	:	0415
			B4 AA B4	AA 00067			:	
			4C15	8F B0 00069	MOVW	#19477, \$RMS_PTR	:	
			C7 AA	02 90 0006F	MOVB	#2, \$RMS_PTR+19	:	
0058	8F	00	E2 AA	02 90 00073	MOVB	#2, \$RMS_PTR+46	:	
			6E	00 2C 00077	MOVCS	#0, (SP), #0, #88, \$RMS_PTR	:	0423
			6A 5813	6A 0007E			:	
			04 AA B4	8F B0 0007F	MOVW	#22547, \$RMS_PTR	:	
			08 AA FF00	AA 9E 00084	MOVAB	KEYXAB, \$RMS_PTR+4	:	
			0C AA 00010004	8F B0 00089	MOVW	#-256, \$RMS_PTR+8	:	
				8F D0 0008F	MOVL	#6554, \$RMS_PTR+12	:	
			00000000G	56 DD 00097	PUSHL	FAB	:	0425
				01 FB 00099	CALLS	#1, SYS\$CREATE	:	
				34 11 000A0	BRB	3\$:	0389
0050	8F	00	6E	00 2C 000A2 2\$:	MOVCS	#0, (SP), #0, #80, (FAB)	:	0440
				66 000A9			:	
			66 5003	8F B0 000AA	MOVW	#20483, (FAB)	:	
			0F02	8F B0 000AF	MOVW	#3842, 22(FAB)	:	
			16 A6	02 90 000B5	MOVB	#2, 31(FAB)	:	
			1F A6	58 D0 000B9	MOVL	FNA, 44(FAB)	:	
			2C A6	00 9E 000BD	MOVAB	P.AAT, 48(FAB)	:	
			30 A6 00000000'	59 90 000C5	MOVB	FNS, 52(FAB)	:	
			34 A6	0F 90 000C9	MOVB	#15, 53(FAB)	:	
			35 A6	56 DD 000CD	PUSHL	FAB	:	0442
			00000000G	01 FB 000CF	CALLS	#1, SYS\$OPEN	:	
				50 E9 000D6 3\$:	BLBC	STATUS, 4\$:	0446
0044	8F	00	6E	00 2C 000D9	MOVCS	#0, (SP), #0, #68, (RAB)	:	0459
				67 000E0			:	
			67 4401	8F B0 000E1	MOVW	#17409, (RAB)	:	
			00200010	8F D0 000E6	MOVL	#2097168, 4(RAB)	:	
			04 A7	01 90 000EE	MOVB	#1, 30(RAB)	:	
			1E A7	CA 9E 000F2	MOVAB	NMA\$W KEYBUF, 48(RAB)	:	
			30 A7 FB10	02 90 000F8	MOVB	#2, 52(RAB)	:	
			34 A7	56 D0 000FC	MOVL	FAB, 60(RAB)	:	
			3C A7	57 DD 00100	PUSHL	RAB	:	0461
			00000000G	01 FB 00102	CALLS	#1, SYS\$CONNECT	:	
				04 00109 4\$:	RET		:	0463

; Routine Size: 266 bytes, Routine Base: \$CODE\$ + 00DE


```

469 0464 1 %SBTTL 'NMA$CLOSEFILE Close a specified file'
470 0465 1 GLOBAL ROUTINE NMA$CLOSEFILE (FILEID) =
471 0466 1
472 0467 1 |++
473 0468 1 | FUNCTIONAL DESCRIPTION:
474 0469 1 |
475 0470 1 |     This routine closes a specified file or all the files.
476 0471 1 |
477 0472 1 | FORMAL PARAMETERS:
478 0473 1 |
479 0474 1 |     FILEID          Value of the fileid parameter (NMA$OPN_xxxxx)
480 0475 1 |
481 0476 1 | ROUTINE VALUE:
482 0477 1 | COMPLETION CODES:
483 0478 1 |
484 0479 1 |     Status of last close operation.
485 0480 1 |
486 0481 1 | --
487 0482 1
488 0483 2 BEGIN
489 0484 2
490 0485 2 LOCAL
491 0486 2     FAB : REF BLOCK [1, BYTE],           ! The fab for the file
492 0487 2     FILEDSC : REF BLOCK [1, BYTE],      ! File descriptor
493 0488 2     FIELD (FDSCFLDS),
494 0489 2     STATUS;                               ! Status return
495 0490 2
496 0491 2 STATUS = NMA$SUCCESS;
497 0492 2 IF NMA$SELECTFILE (.FILEID, FILEDSC) THEN ! Obtain descriptor address
498 0493 3     BEGIN
499 0494 3     FAB = .FILEDSC [FDSCFAB];             ! Get address of FAB
500 0495 3     IF .FAB [FABSW_IFI] NEQ 0 THEN        ! If file isn't closed, do it.
501 0496 4     BEGIN
502 0497 4     STATUS =
503 0498 4     $CLOSE (FAB = .FILEDSC [FDSCFAB]); ! Call RMS to close the file
504 0499 4     IF .STATUS THEN
505 0500 4     NML$LOGFILEOP (DBG$C FILEID,
506 0501 4     FILEID,
507 0502 4     $ASCID ('file closed.));
508 0503 3     END;
509 0504 3     END
510 0505 2 ELSE
511 0506 2     STATUS = NMA$_BADFILEID;
512 0507 2     RETURN .STATUS
513 0508 2
514 0509 1 END;

```

```

.PSECT $SPLITS, NOWRT, NOEXE, 2
2E 64 65 73 6F 6C 63 20 65 6C 69 66 000FA P.AAV: .ASCII \file closed.\
                                00106          .BLKB 2
                                0000000C 00108 P.AAU: .LONG 12
                                00000000' 0010C .ADDRESS P.AAV
                                .EXTRN SYS$CLOSE

```

				.PSECT	\$CODE\$,NOWRT,2	
			0004 000C0	.ENTRY	NMA\$CLOSEFILE, Save R2	: 0465
	5E		04 C2 00002	SUBL2	#4, SP	: 0491
	52		01 D0 00005	MOVL	#1, STATUS	: 0492
			5E DD 00008	PUSHL	SP	: 0494
		04	AC DD 0000A	PUSHL	FILEID	: 0495
FE87	CF		02 FB 0000D	CALLS	#2, NMA\$SELECTFILE	: 0496
	30		50 E9 00012	BLBC	R0, 1\$: 0498
	50		6E D0 00015	MOVL	FILEID, R0	: 0499
	51	08	A0 D0 00018	MOVL	8(R0), FAB	: 0500
		02	A1 B5 0001C	TSTW	2(FAB)	: 0501
			26 13 0001F	BEQL	2\$: 0502
		08	A0 DD 00021	PUSHL	8(R0)	: 0503
00000000G	00		01 FB 00024	CALLS	#1, SYSS\$CLOSE	: 0504
	52		50 D0 00028	MOVL	R0, STATUS	: 0505
	16		52 E9 0002E	BLBC	STATUS, 2\$: 0506
			00 9F 00031	PUSHAB	P.AAJ	: 0507
		04	AC DC 00037	PUSHL	FILEID	: 0508
			01 DD 0003A	PUSHL	#1	: 0509
00000000G	00		03 FB 0003C	CALLS	#3, NML\$LOGFILEOP	: 0492
			02 11 00043	BRB	2\$: 0506
			52 D4 00045 1\$:	CLRL	STATUS	: 0507
	50		52 D0 00047 2\$:	MOVL	STATUS, R0	: 0507
			04 0004A	RET		: 0509

: Routine Size: 75 bytes, Routine Base: \$CODE\$ + 01E8

```

516 0510 1 %SBTTL 'NMASMATCHREC Find a Record in a File'
517 0511 1 GLOBAL ROUTINE NMASMATCHREC (FILEID, BUFDSC, KEYADR, FIELD CODE,
518 0512 1 FIELD SIZE, FIELDADR, RTNDSC) =
519 0513 1
520 0514 1 ++
521 0515 1 FUNCTIONAL DESCRIPTION:
522 0516 1
523 0517 1 This routine searches a database for a record containing a given
524 0518 1 field containing given data. Degenerate cases are provided for
525 0519 1 returning all records, or all records containing a specific field.
526 0520 1
527 0521 1 FORMAL PARAMETERS:
528 0522 1
529 0523 1 FILEID Value of the fileid code (NMASC OPN_xxxxx)
530 0524 1 BUFDSC Address of a descriptor of a buffer to use
531 0525 1 KEYADR Address of a word containing the key to start reading
532 0526 1 Key value is returned in this word.
533 0527 1 FIELD CODE Value of the field code (zero for wildcard)*****
534 0528 1 FIELD SIZE Value of the field size (zero for wildcard)
535 0529 1 FIELDADR Address of the field data
536 0530 1 RTNDSC Address of a descriptor to return descriptor of data
537 0531 1
538 0532 1 IMPLICIT INPUTS:
539 0533 1
540 0534 1 NONE
541 0535 1
542 0536 1 IMPLICIT OUTPUTS:
543 0537 1
544 0538 1 NONE
545 0539 1
546 0540 1 ROUTINE VALUE:
547 0541 1 COMPLETION CODES.
548 0542 1
549 0543 1 NMA or RMS error status
550 0544 1
551 0545 1 SIDE EFFECTS:
552 0546 1
553 0547 1 NONE
554 0548 1
555 0549 1 --
556 0550 1
557 0551 2 BEGIN
558 0552 2
559 0553 2 MAP
560 0554 2 BUFDSC : REF VECTOR, ! Buffer to use for record
561 0555 2 RTNDSC : REF VECTOR; ! Return data descriptor
562 0556 2
563 0557 2 LOCAL
564 0558 2 FILEDSC : REF BLOCK [1, BYTE] ! File descriptor
565 0559 2 FIELD (FDSCFLDS)
566 0560 2 RAB : REF BLOCK [1, BYTE], ! The rab for the file
567 0561 2 LCLDSC : VECTOR [2], ! A local data descriptor
568 0562 2 FAB : REF BLOCK [, BYTE], ! The fab for the file
569 0563 2 FLDADR, ! Field address
570 0564 2 FLDSIZ, ! Field size
571 0565 2 STATUS; ! Status return
572 0566 2

```

```

573 0567 2 EXTERNAL ROUTINE
574 0568 2 NMA$SEARCHFLD; ! Search for a field value
575 0569 2
576 0570 2 STATUS = NMA$SELECTFILE (.FILEID,
577 0571 2 FILEDSC); ! Obtain the file descriptor
578 0572 2
579 0573 2 IF NOT .STATUS
580 0574 2 THEN
581 0575 2 RETURN .STATUS; ! Bogus fileid
582 0576 2
583 0577 2 RAB = .FILEDSC [FDSCRAB]; ! Point to the rab
584 0578 2 FAB = .FILEDSC [FDSCFAB]; ! Get address of FAB
585 0579 2
586 0580 2 IF .FAB [FAB$W_IFI] EQL 0 ! If file not open,
587 0581 2 THEN
588 0582 2 RETURN .FAB [FAB$L_STS]; ! return open failure status
589 0583 2
590 0584 2 RAB [RAB$W_USZ] = .BUFDSC [0]; ! Set the buffer to use
591 0585 2 RAB [RAB$L_UBF] = .BUFDSC [1];
592 0586 2
593 0587 2 NMA$W_KEYBUF = ..KEYADR; ! And the key value to use
594 0588 2
595 0589 2 WHILE 1 ! Try this forever
596 0590 2 DO
597 0591 2 BEGIN
598 0592 2
599 0593 2 STATUS = $GET (RAB = .RAB); ! Read a record
600 0594 2
601 0595 2 LCLDSC [0] = .RAB [RAB$W_RSZ]; ! Pickup the real record descriptor
602 0596 2 LCLDSC [1] = .RAB [RAB$L_RBF];
603 0597 2 RTNDSC [0] = .RAB [RAB$W_RSZ] - NML$K_PERM_KEYS_LEN;
604 0598 2 RTNDSC [1] = .RAB [RAB$L_RBF] + NML$K_PERM_KEYS_LEN;
605 0599 2
606 0600 2 IF NOT .STATUS ! If no good, return
607 0601 2 THEN
608 0602 2 RETURN .STATUS;
609 0603 2
610 0604 2 NMA$W_KEYBUF = ! Set the keyvalue returned
611 0605 2 (.LCLDSC [1]) <0, 16, 0>;
612 0606 2
613 0607 2 (.KEYADR) <0, 16, 0> = .NMA$W_KEYBUF; ! Return for user to remember
614 0608 2
615 0609 2 FLDADR = 0; ! Start search from beginning
616 0610 2 IF NMA$SEARCHFLD ! Look for the field
617 0611 2 (
618 0612 2 .RTNDSC, ! Here is the data
619 0613 2 .FIELD$CODE, ! Value of the code to look for
620 0614 2 FLDSIZ, ! Return the size here
621 0615 2 FLDADR ! Return the address here
622 0616 2 )
623 0617 2 THEN
624 0618 2 BEGIN
625 0619 2
626 0620 2 IF .FIELD$SIZE EQL 0 ! Wildcard
627 0621 2 THEN
628 0622 2 BEGIN
629 0623 2

```

```

: 630      0624      5          STATUS = NMA$_SUCCESS; ! It always succeeds
: 631      0625      5          EXITLOOP;
: 632      0626      5
: 633      0627      4          END;
: 634      0628      4
: 635      0629      4          IF CH$EQL          ! Look at the data
: 636      0630      4          (
: 637      0631      4          .FLDSIZ,          ! Data in record
: 638      0632      4          .FLDADR,
: 639      0633      4          .FIELD$IZE,      ! User data
: 640      0634      4          .FIELDADR,
: 641      0635      4          0
: 642      0636      4          )
: 643      0637      4          THEN
: 644      0638      5          BEGIN
: 645      0639      5
: 646      0640      5          STATUS = NMA$_SUCCESS; ! We found such a record
: 647      0641      5          EXITLOOP;
: 648      0642      5
: 649      0643      4          END;
: 650      0644      3          END;
: 651      0645      3
: 652      0646      3          NMA$W_KEYBUF = .NMA$W_KEYBUF + 1; ! Increment key ****
: 653      0647      3          (.KEYADR) <0, 16, 0> = .NMA$W_KEYBUF; ! Return for user to remember
: 654      0648      3
: 655      0649      2          END;
: 656      0650      2
: 657      0651      2          IF .STATUS
: 658      0652      2          THEN
: 659      0653      2          NML$LOGRECORDOP (DBG$C_FILEID,
: 660      0654      2          .FILEID,
: 661      0655      2          $ASCID ('record matched'),
: 662      0656      2          LC(DSC));
: 663      0657      2
: 664      0658      2          RETURN .STATUS
: 665      0659      2
: 666      0660      1          END;

```

.PSECT \$PLITS, NOWRT, NOEXE, 2

```

64 65 68 63 74 61 6D 20 64 72 6F 63 65 72 00110 P.AAX: .ASCII \record matched\      ;
:                                     0011E .BLKB 2 ;
:                                     0000000E 00120 P.AAW: .LONG 14 ;
:                                     00000000' 00124 .ADDRESS P.AAX ;
:
: .EXTRN NMA$SEARCHFLD, SYSSGET
:
: .PSECT $CODE$, NOWRT, 2
:
: .ENTRY NMA$MATCHREC, Save R2,R3,R4,R5,R6,R7      ; 0511
: MOVAB NMA$W_KEYBUF, R7
: SUBL2 #20, SP
: PUSHL SP ; 0570
: PUSHL FILEID
: CALLS #2, NMA$SELECTFILE

```

```

FE38 CF          04 AC DD 0000E
: 57 00000000' 00 9E 00002
: SE          14 C2 00009
:          5E DD 0000C
:          02 FB 00011

```

		56		50	D0	00016	MOVL	R0, STATUS				
		4F		56	E9	00019	BLBC	STATUS, 3\$	0573			
		50		6E	D0	0001C	MOVL	FILEDSC, R0	0577			
		54	0C	A0	D0	0001F	MOVL	12(R0), RAB				
		50	08	A0	D0	00023	MOVL	8(R0), FAB	0578			
			02	A0	B5	00027	TSTW	2(FAB)	0580			
				05	12	0002A	BNEQ	1\$				
		50	08	A0	D0	0002C	MOVL	8(FAB), R0	0582			
				04	00	00030	RET					
		50	08	AC	D0	00031	1\$: MOVL	BUFDSC, R0	0584			
20		A4		60	B0	00035	MOVW	(R0), 32(RAB)				
24		A4	04	A0	D0	00039	MOVL	4(R0), 36(RAB)	0585			
		67	0C	BC	B0	0003E	MOVW	@KEYADR, NMA\$W_KEYBUF	0587			
		55	1C	AC	D0	00042	MOVL	RTNDSC, R5	0598			
				54	DD	00046	2\$: PUSHL	RAB	0593			
	00000000G	00		01	FB	00048	CALLS	#1, SYSSGET				
		56		50	D0	0004F	MOVL	R0, STATUS				
	0C	AE	22	A4	3C	00052	MOVZWL	34(RAB), LCLDSC	0595			
	10	AE	28	A4	D0	00057	MOVL	40(RAB), LCLDSC+4	0596			
	1C	BC	22	A4	3C	0005C	MOVZWL	34(RAB), @RTNDSC	0597			
	1C	BC		02	C2	00061	SUBL2	#2, @RTNDSC				
04	A5	28		02	C1	00065	ADDL3	#2, 40(RAB), 4(R5)	0598			
		57		56	E9	0006B	3\$: BLBC	STATUS, 7\$	0600			
		67	10	BE	B0	0006E	MOVW	@LCLDSC+4, NMA\$W_KEYBUF	0605			
		BC		67	B0	00072	MOVW	NMA\$W_KEYBUF, @KEYADR	0607			
				04	AE	D4	00076	CLRL	FLDADR	0609		
				04	AE	9F	00079	PUSHAB	FLDADR	0611		
				0C	AE	9F	0007C	PUSHAB	FLDSIZ			
				10	AC	DD	0007F	PUSHL	FIELDLDR	0613		
				1C	AC	DD	00082	PUSHL	RTNDSC	0612		
	00000000G	00		04	FB	00085	CALLS	#4, NMA\$SEARCHFLD				
		16		50	E9	0008C	BLBC	R0, 5\$				
				14	AC	D5	0008F	TSTL	FIELDLDR	0620		
				0C	13	00092	BEQL	4\$				
14	AC		00	04	BE	08	AE	2D	00094	CMPCS	FLDSIZ, @FLDADR, #0, FIELDLDR, @FIELDADR	0630
				18	BC				0009C			
				05	12	0009E	BNEQ	5\$				
		56		01	D0	000A0	4\$: MOVL	#1, STATUS	0640			
				08	11	000A3	BRB	6\$	0638			
				67	B6	000A5	5\$: INCW	NMA\$W_KEYBUF	0646			
		0C	BC	67	B0	000A7	MOVW	NMA\$W_KEYBUF, @KEYADR	0647			
				99	11	000AB	BRB	2\$	0589			
		15		56	E9	000AD	6\$: BLBC	STATUS, 7\$	0651			
			0C	AE	9F	000B0	PUSHAB	LCLDSC	0653			
			00000000	00	9F	000B3	PUSHAB	P.AAW	0655			
			04	AC	DD	000B9	PUSHL	FILEID	0654			
				01	DD	000BC	PUSHL	#1	0653			
	00000000G	00		04	FB	000BE	CALLS	#4, NML\$LOGRECORDOP				
		50		56	D0	000C5	7\$: MOVL	STATUS, R0	0658			
				04	00	000C8	RET		0660			

; Routine Size: 201 bytes. Routine Base: \$CODE\$ + 0233

```

668 0661 1 XSBTTL 'NMA$READREC Get a record from a File'
669 0662 1 GLOBAL ROUTINE NMA$READREC (FILEID, KEYADR, BUFDSC, RTNDSC) =
670 0663 1
671 0664 1 |++
672 0665 1 | FUNCTIONAL DESCRIPTION:
673 0666 1 |
674 0667 1 |     This routine reads the next database record starting at the specified
675 0668 1 |     key.
676 0669 1 |
677 0670 1 | FORMAL PARAMETERS:
678 0671 1 |
679 0672 1 |     FILEID      Value of the fileid code (NMASC_OPN_xxxxx)
680 0673 1 |     KEYADR      Address of a word containing the key to start reading
681 0674 1 |                 Key value is returned in this word.
682 0675 1 |     BUFDSC      Address of a descriptor of a buffer to use
683 0676 1 |     RTNDSC      Address of a descriptor to return descriptor of data
684 0677 1 |
685 0678 1 | IMPLICIT INPUTS:
686 0679 1 |
687 0680 1 |     NONE
688 0681 1 |
689 0682 1 | IMPLICIT OUTPUTS:
690 0683 1 |
691 0684 1 |     NONE
692 0685 1 |
693 0686 1 | ROUTINE VALUE:
694 0687 1 | COMPLETION CODES:
695 0688 1 |
696 0689 1 |     NMA or RMS error status
697 0690 1 |
698 0691 1 | SIDE EFFECTS:
699 0692 1 |
700 0693 1 |     NONE
701 0694 1 |
702 0695 1 | --
703 0696 1 |
704 0697 2 | BEGIN
705 0698 2 |
706 0699 2 | MAP
707 0700 2 |     BUFDSC : REF VECTOR,           ! Buffer to use for record
708 0701 2 |     RTNDSC : REF VECTOR;          ! Return data descriptor
709 0702 2 |
710 0703 2 | LOCAL
711 0704 2 |     FILEDSC : REF BLOCK [1, BYTE] ! File descriptor
712 0705 2 |     FIELD (FDSCFLDS),
713 0706 2 |     FAB      : REF BLOCK [1, BYTE], ! The fab for the file
714 0707 2 |     RAB      : REF BLOCK [1, BYTE], ! The rab for the file
715 0708 2 |     LCLDSC  : VECTOR [2],
716 0709 2 |     STATUS;                          ! Status return
717 0710 2 |
718 0711 2 | STATUS = NMA$SELECTFILE (.FILEID,
719 0712 2 |                        FILEDSC);    ! Obtain the file descriptor
720 0713 2 |
721 0714 2 | IF NOT .STATUS
722 0715 2 | THEN
723 0716 2 |     RETURN .STATUS;                ! Bogus fileid
724 0717 2 |

```

```

: 725 0718 2
: 726 0719 2 RAB = .FILEDSC [FDSCRAB]; ! Point to the rab
: 727 0720 2 FAB = .FILEDSC [FDSCFAB]; ! Get address of FAB
: 728 0721 2
: 729 0722 2 IF .FAB [FAB$J_IFI] EQL 0 ! If file not open,
: 730 0723 2 THEN
: 731 0724 2 RETURN .FAB [FAB$L_STS]; ! Return open failure status
: 732 0725 2
: 733 0726 2 RAB [RAB$W_USZ] = .BUFDSC [0]; ! Set the buffer to use
: 734 0727 2 RAB [RAB$L_UBF] = .BUFDSC [1];
: 735 0728 2
: 736 0729 2 NMA$W_KEYBUF = ..KEYADR; ! And the key value to use
: 737 0730 2
: 738 0731 2 STATUS = $GET (RAB = .RAB); ! Read a record
: 739 0732 2
: 740 0733 2 RTNDSC [0] = .RAB [RAB$W_RSZ] - NML$K_PERM_KEYS_LEN;
: 741 0734 2 RTNDSC [1] = .RAB [RAB$L_RBF] + NML$K_PERM_KEYS_LEN;
: 742 0735 2
: 743 0736 2 IF NOT .STATUS ! If no good, return
: 744 0737 2 THEN
: 745 0738 2 RETURN .STATUS;
: 746 0739 2
: 747 0740 2 LCLDSC [0] = .RAB [RAB$W_RSZ];
: 748 0741 2 LCLDSC [1] = .RAB [RAB$L_RBF];
: 749 0742 2
: 750 0743 2 (.KEYADR)<0,16,0> = (.LCLDSC [1])<0,16>; ! Return for user to remember
: 751 0744 2
: 752 0745 2 NML$LOGRECORDOP (DBG$C_FILEID,
: 753 0746 2 .FILEID,
: 754 0747 2 $ASCID ('record read'),
: 755 0748 2 LCLDSC);
: 756 0749 2
: 757 0750 2 RETURN NMA$_SUCCESS
: 758 0751 2
: 759 0752 1 END;

```

```

.PSECT $PLITS,NOWRT,NOEXE,2
64 61 65 72 20 64 72 6F 63 65 72 00128 P.AAZ: .ASCII \record read\ :
00133 .BLKB 1 :
0000000B 00134 P.AAY: .LONG 11 :
00000000 00138 .ADDRESS P.AAZ :

.PSECT $CODE$,NOWRT,2
SE 0004 0000 .ENTRY NMA$READREC, Save R2 : 0662
OC C2 00002 SUBL2 #12, SP :
SE DD 00005 PUSHL SP : 0711
04 AC DD 00007 PUSHL FILEID :
FD76 CF 02 FB 0000A CALLS #2, NMA$SELECTFILE :
6A 50 E9 0000F BLBC STATUS, 2$ : 0714
51 6E D0 00012 MOVL FILEDSC, R1 : 0719
51 08 A1 7D 00015 MOVQ 8(R1), FAB : 0720

```


			02	A1	B5	00019	TSTW	2(FAB)	:	0722
			05	12	0001C		BNEQ	1\$:	
		50	08	A1	D0	0001E	MOVL	8(FAB), R0	:	0724
					04	00022	RET		:	
		51	0C	AC	D0	00023	MOVL	BUFDSC, R1	:	0726
	20	A2		61	B0	00027	MOVW	(R1), 32(RAB)	:	
	24	A2	04	A1	D0	0002B	MOVL	4(R1), 36(RAB)	:	0727
	00000000'	00	08	BC	B0	00030	MOVW	@KEYADR, NMA\$W_KEYBUF	:	0729
				52	DD	00038	PUSHL	RAB	:	0731
	00000000G	00		01	FB	0003A	CALLS	#1, SYSSGET	:	
		51	10	AC	D0	00041	MOVL	RTNDSC, R1	:	0733
		61	22	A2	3C	00045	MOVZWL	34(RAB), (R1)	:	
		61		02	C2	00049	SUBL2	#2, (R1)	:	
04	A1	28		02	C1	0004C	ADDL3	#2, 40(RAB), 4(R1)	:	0734
		27		50	E9	00052	BLBC	STATUS, 2\$:	0736
		04	22	A2	3C	00055	MOVZWL	34(RAB), LCLDSC	:	0740
		08	28	A2	D0	0005A	MOVL	40(RAB), LCLDSC+4	:	0741
		08	08	BE	B0	0005F	MOVW	@LCLDSC+4, @KEYADR	:	0743
			04	AE	9F	00064	PUSHAB	LCLDSC	:	0745
			00000000'	00	9F	00067	PUSHAB	P.AAY	:	0747
			04	AC	DD	0006D	PUSHL	FILEID	:	0746
				01	DD	00070	PUSHL	#1	:	0745
	00000000G	00		04	FB	00072	CALLS	#4, NML\$LOGRECORDOP	:	
		50		01	D0	00079	MOVL	#1, R0	:	0750
				04	0007C	2\$:	RET		:	0752

; Routine Size: 125 bytes, Routine Base: \$CODE\$ + 02FC

```

761 0753 1 %SBTTL 'NMA$WRITEREC Write a Record to a File'
762 0754 1 GLOBAL ROUTINE NMA$WRITEREC (FILEID, KEYADR, BUFDSC) =
763 0755 1
764 0756 1 |++
765 0757 1 | FUNCTIONAL DESCRIPTION:
766 0758 1 |
767 0759 1 |     This routine puts a record to the specified file. The key is
768 0760 1 |     specified by keyadr. The file was opened so that puts to existing
769 0761 1 |     records act as updates. The keyvalue is moved to the first two bytes
770 0762 1 |     of the record before the write.
771 0763 1 |
772 0764 1 | FORMAL PARAMETERS:
773 0765 1 |
774 0766 1 |     FILEID      Value if the fileid
775 0767 1 |     KEYADR      Address of a word of keyvalue
776 0768 1 |     BUFDSC      Address of descriptor of data to write
777 0769 1 |
778 0770 1 | IMPLICIT INPUTS:
779 0771 1 |
780 0772 1 |     NONE
781 0773 1 |
782 0774 1 | IMPLICIT OUTPUTS:
783 0775 1 |
784 0776 1 |     NONE
785 0777 1 |
786 0778 1 | ROUTINE VALUE:
787 0779 1 | COMPLETION CODES:
788 0780 1 |
789 0781 1 |     RMS error code
790 0782 1 |
791 0783 1 | SIDE EFFECTS:
792 0784 1 |
793 0785 1 |     NONE
794 0786 1 |
795 0787 1 | --
796 0788 1 |
797 0789 2 | BEGIN
798 0790 2 |
799 0791 2 | MAP
800 0792 2 |     BUFDSC : REF VECTOR;           ! User supplied data
801 0793 2 |
802 0794 2 | LOCAL
803 0795 2 |     RAB      : REF BLOCK [1, BYTE], ! Address of rab
804 0796 2 |     STATUS   : REF BLOCK [1, BYTE], ! Return status
805 0797 2 |     FILEDSC  : REF BLOCK [1, BYTE]  ! File descriptor address
806 0798 2 |     FIELD (FDSCFLDS),
807 0799 2 |     LCLDSC   : VECTOR [2];
808 0800 2 |
809 0801 2 |     STATUS = NMA$SELECTFILE (.FILEID,
810 0802 2 |                             FILEDSC); ! Obtain file descriptor
811 0803 2 | IF NOT .STATUS
812 0804 2 | THEN
813 0805 2 |     RETURN .STATUS;               ! Return the status
814 0806 2 |
815 0807 2 |     RAB = .FILEDSC [FDSCRAB];      ! Obtain the rab address
816 0808 2 |     LCLDSC [0] = .BUFDSC [0] + NML$K_PERM_KEYS_LEN;
817 0809 2 |     LCLDSC [1] = .BUFDSC [1] - NML$K_PERM_KEYS_LEN;

```

```

: 818 0810 2 RAB [RAB$W_RSZ] = .LCLDSC [0]; ! User buffer to write
: 819 0811 2 RAB [RAB$S_RBF] = .LCLDSC [1];
: 820 0812 2
: 821 0813 2 NMA$W_KEYBUF = .KEYADR; ! Key value from user
: 822 0814 2 (.LCLDSC [1])<0,16,0> = .NMA$W_KEYBUF; ! Move key to buffer for write
: 823 0815 2
: 824 0816 2 STATUS = $PUT (RAB = .RAB); ! Put or update the record
: 825 0817 2
: 826 0818 2 IF .STATUS
: 827 0819 2 THEN
: 828 0820 2 NML$LOGRECORDOP (DBG$C FILEID,
: 829 0821 2 FILEID,
: 830 0822 2 $ASCID ('record written'),
: 831 0823 2 LCLDSC);
: 832 0824 2
: 833 0825 2 RETURN .STATUS
: 834 0826 2
: 835 0827 1 END;

```

```

.PSECT $SPLITS,NOWRT,NOEXE,2
6E 65 74 74 69 72 77 20 64 72 6F 63 65 72 0013C P.ABB: .ASCII \record written\
0014A .BLKB 2
0000000E 0014C P.ABA: .LONG 14
00000000' 00150 .ADDRESS P.ABB

.EXTRN SYSSPUT

.PSECT $CODE$,NOWRT,2

.ENTRY NMA$WRITEREC, Save R2,R3 : 0754
MOVAB NMA$W_KEYBUF, R3
SUBL2 #12, SP
PUSHL SP : 0801
PUSHL FILEID
CALLS #2, NMA$SELECTFILE
MOVL R0, STATUS
BLBC STATUS, 1$ : 0803
MOVL FILEDSC, R0 : 0807
MOVL 12(R0), RAB
MOVL BUFDC, R0 : 0808
ADDL3 #2, (R0), LCLDSC
SUBL3 #2, 4(R0), LCLDSC+4 : 0809
MOVW LCLDSC, 34(RAB) : 0810
MOVL LCLDSC+4, 40(RAB) : 0811
MOVW @KEYADR, NMA$W_KEYBUF : 0813
MOVW NMA$W_KEYBUF, 3LCLDSC+4 : 0814
PUSHL RAB : 0816
CALLS #1, SYSSPUT
MOVL R0, STATUS
BLBC STATUS, 1$ : 0818
PUSHAB LCLDSC : 0820
PUSHAB P.ABA : 0822
PUSHL FILEID : 0821
PUSHL #1 : 0820

```

NMAFILES
V04-000

File Routines for Network Management
NMA\$WRITEREC Write a Record to a File

L 1
16-Sep-1984 00:42:37
14-Sep-1984 12:50:02

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

00000000G 00
50

04 FB 00061
52 D0 00068 1s:
04 0006B

CALLS #4, NML\$LOGRECORDOP
MOVL STATUS, R0
RFT

:
: 0825
: 0827

: Routine Size: 108 bytes, Routine Base: \$CODE\$ + 0379

NML
V04

: R

```

837 0828 1 %SBTTL 'NMA$DELETEREC Delete a Record from the File'
838 0829 1 GLOBAL ROUTINE NMA$DELETEREC (FILEID, KEYADR) =
839 0830 1
840 0831 1 ++
841 0832 1 FUNCTIONAL DESCRIPTION:
842 0833 1
843 0834 1     This routine deletes a record from the file by specified key
844 0835 1     number.
845 0836 1
846 0837 1 FORMAL PARAMETERS:
847 0838 1
848 0839 1     FILEID      Value if the fileid
849 0840 1     KEYADR      Address of a word of keyvalue
850 0841 1
851 0842 1 IMPLICIT INPUTS:
852 0843 1
853 0844 1     NONE
854 0845 1
855 0846 1 IMPLICIT OUTPUTS:
856 0847 1
857 0848 1     NONE
858 0849 1
859 0850 1 ROUTINE VALUE:
860 0851 1 COMPLETION CODES:
861 0852 1
862 0853 1     RMS error code
863 0854 1
864 0855 1 SIDE EFFECTS:
865 0856 1
866 0857 1     NONE
867 0858 1
868 0859 1 --
869 0860 1
870 0861 2 BEGIN
871 0862 2
872 0863 2 LOCAL
873 0864 2     RAB      : REF BLOCK [1, BYTE], ! Address of rab
874 0865 2     STATUS,  ! Return status
875 0866 2     FILEDSC : REF BLOCK [1, BYTE] ! File descriptor address
876 0867 2     FIELD (FDSCFLDS);
877 0868 2
878 0869 2     STATUS = NMA$SELECTFILE (.FILEID,
879 0870 2     FILEDSC); ! Obtain file descriptor
880 0871 2
881 0872 2 IF .STATUS
882 0873 2 THEN
883 0874 2 BEGIN
884 0875 2
885 0876 2     RAB = .FILEDSC [FDSCRAB]; ! Obtain the rab address
886 0877 2
887 0878 2     NMA$W_KEYBUF = ..KEYADR; ! Key value from user
888 0879 2
889 0880 2     STATUS = $DELETE (RAB = .RAB); ! Delete the record
890 0881 2
891 0882 2 IF .STATUS
892 0883 2 THEN
893 0884 2     NML$LOGRECORDOP (DBG$C_FILEIO,

```

```

: 894      0885  3
: 895      0886  3
: 896      0887  3
: 897      0888  3
: 898      0889  3
: 899      0890  3
: 900      0891  3
: 901      0892  3
: 902      0893  1

```

END;
RETURN .STATUS
END;

```

.FILEID,  

$ASCII ('record deleted'),  

UPLIT (2, NMA$W_KEYBUF));

```

```

.PSECT $SPLITS,NOWRT,NOEXE,2
64 65 74 65 6C 65 64 20 64 72 6F 63 65 72 00154 P.ABD: .ASCII \record deleted\
00162 .BLKB 2
0000000E 00164 P.ABC: .LONG 14
00000000' 00168 .ADDRESS P.ABD
00000002 0016C P.ABE: .LONG 2
00000000' 00170 .ADDRESS NMA$W_KEYBUF

.EXTRN SYSS$DELETE

.PSECT $CODE$,NOWRT,2
0004 00000 .ENTRY NMA$DELETEREC, Save R2
5E 04 C2 00002 SUBL2 #4, SP
5E DD 00005 PUSHL SP
04 AC DD 00007 PUSHL FILEID
FC8D CF 02 FB 0000A CALLS #2, NMA$SELECTFILE
52 50 D0 0000F MOVL R0, STATUS
36 52 E9 00012 BLBC STATUS, 1$
50 6E D0 00015 MOVL FILEDSC, R0
50 0C A0 D0 00018 MOVL 12(R0), RAB
00000000' 00 08 BC B0 0001C MOVW @KEYADR, NMA$W_KEYBUF
00000000G 00 01 FB 00026 PUSHL RAB
52 50 D0 0002D CALLS #1, SYSS$DELETE
18 52 E9 00030 MOVL R0, STATUS
00000000' 00 9F 00033 BLBC STATUS, 1$
00000000' 00 9F 00039 PUSHAB P.ABE
04 AC DD 0003F PUSHAB P.ABC
01 DD 00042 PUSHL FILEID
00000000G 00 04 FB 00044 PUSHL #1
50 52 D0 0004B CALLS #4, NML$LOGRECORDOP
04 0004E MOVL STATUS, R0
RET
1$:
: 0829
: 0869
: 0872
: 0876
: 0878
: 0880
: 0882
: 0887
: 0886
: 0885
: 0884
: 0891
: 0893

```

; Routine Size: 79 bytes, Routine Base: \$CODE\$ + 03E5

NMAFILES
V04-000

File Routines for Network Management
NMA\$DELETEREC Delete a Record from the File

B 2
16-Sep-1984 00:42:37
14-Sep-'984 12:50:02

VAX-11 Bliss-32 V4.0-742
DISK\$VM\$MASTER:[NML.SRC]NMAFILES.B32;1 (11) Page 29

NML
V04

: 904 0894 1 END
: 905 0895 1
: 906 0896 0 ELUDOM

. End of module

: R

PSECT SUMMARY

Name	Bytes	Attributes
\$OWNS	1352	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$PLITS	372	NOVEC, NOWRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$CODES	1076	NOVEC, NOWRT, RD, FXE, 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	3	0	27	00:00.1
-\$255\$DUA28:[SHRLIB]NMLIBRY.L32;1	887	14	1	47	00:00.2
-\$255\$DUA28:[SYSLIB]STARLET.L32;1	9776	141	1	581	00:02.2

COMMAND QUALIFIERS

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

: Size: 1076 code + 1724 data bytes
: Run Time: 00:30.1
: Elapsed Time: 01:12.0
: Lines/CPU Min: 1784
: Lexemes/CPU-Min: 31149
: Memory Used: 196 pages
: Compilation Complete

001	002	003	004	005	006	007	008	009	010	011	012	013	014	015	016	017	018	019	020	021	022	023	024	025	026	027	028	029	030	031	032	033	034	035	036	037	038	039	040	041	042	043	044	045	046	047	048	049	050	051	052	053	054	055	056	057	058	059	060	061	062	063	064	065	066	067	068	069	070	071	072	073	074	075	076	077	078	079	080	081	082	083	084	085	086	087	088	089	090	091	092	093	094	095	096	097	098	099	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255	256	257	258	259	260	261	262	263	264	265	266	267	268	269	270	271	272	273	274	275	276	277	278	279	280	281	282	283	284	285	286	287	288	289	290	291	292	293	294	295	296	297	298	299	300	301	302	303	304	305	306	307	308	309	310	311	312	313	314	315	316	317	318	319	320	321	322	323	324	325	326	327	328	329	330	331	332	333	334	335	336	337	338	339	340	341	342	343	344	345	346	347	348	349	350	351	352	353	354	355	356	357	358	359	360	361	362	363	364	365	366	367	368	369	370	371	372	373	374	375	376	377	378	379	380	381	382	383	384	385	386	387	388	389	390	391	392	393	394	395	396	397	398	399	400	401	402	403	404	405	406	407	408	409	410	411	412	413	414	415	416	417	418	419	420	421	422	423	424	425	426	427	428	429	430	431	432	433	434	435	436	437	438	439	440	441	442	443	444	445	446	447	448	449	450	451	452	453	454	455	456	457	458	459	460	461	462	463	464	465	466	467	468	469	470	471	472	473	474	475	476	477	478	479	480	481	482	483	484	485	486	487	488	489	490	491	492	493	494	495	496	497	498	499	500	501	502	503	504	505	506	507	508	509	510	511	512	513	514	515	516	517	518	519	520	521	522	523	524	525	526	527	528	529	530	531	532	533	534	535	536	537	538	539	540	541	542	543	544	545	546	547	548	549	550	551	552	553	554	555	556	557	558	559	560	561	562	563	564	565	566	567	568	569	570	571	572	573	574	575	576	577	578	579	580	581	582	583	584	585	586	587	588	589	590	591	592	593	594	595	596	597	598	599	600	601	602	603	604	605	606	607	608	609	610	611	612	613	614	615	616	617	618	619	620	621	622	623	624	625	626	627	628	629	630	631	632	633	634	635	636	637	638	639	640	641	642	643	644	645	646	647	648	649	650	651	652	653	654	655	656	657	658	659	660	661	662	663	664	665	666	667	668	669	670	671	672	673	674	675	676	677	678	679	680	681	682	683	684	685	686	687	688	689	690	691	692	693	694	695	696	697	698	699	700	701	702	703	704	705	706	707	708	709	710	711	712	713	714	715	716	717	718	719	720	721	722	723	724	725	726	727	728	729	730	731	732	733	734	735	736	737	738	739	740	741	742	743	744	745	746	747	748	749	750	751	752	753	754	755	756	757	758	759	760	761	762	763	764	765	766	767	768	769	770	771	772	773	774	775	776	777	778	779	780	781	782	783	784	785	786	787	788	789	790	791	792	793	794	795	796	797	798	799	800	801	802	803	804	805	806	807	808	809	810	811	812	813	814	815	816	817	818	819	820	821	822	823	824	825	826	827	828	829	830	831	832	833	834	835	836	837	838	839	840	841	842	843	844	845	846	847	848	849	850	851	852	853	854	855	856	857	858	859	860	861	862	863	864	865	866	867	868	869	870	871	872	873	874	875	876	877	878	879	880	881	882	883	884	885	886	887	888	889	890	891	892	893	894	895	896	897	898	899	900	901	902	903	904	905	906	907	908	909	910	911	912	913	914	915	916	917	918	919	920	921	922	923	924	925	926	927	928	929	930	931	932	933	934	935	936	937	938	939	940	941	942	943	944	945	946	947	948	949	950	951	952	953	954	955	956	957	958	959	960	961	962	963	964	965	966	967	968	969	970	971	972	973	974	975	976	977	978	979	980	981	982	983	984	985	986	987	988	989	990	991	992	993	994	995	996	997	998	999	1000
-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	------

Terminal 1	Terminal 2	Terminal 3	Terminal 4	Terminal 5	Terminal 6	Terminal 7	Terminal 8	Terminal 9	Terminal 10	Terminal 11	Terminal 12
Terminal 13	Terminal 14	Terminal 15	Terminal 16	Terminal 17	Terminal 18	Terminal 19	Terminal 20	Terminal 21	Terminal 22	Terminal 23	Terminal 24
Terminal 25	Terminal 26	Terminal 27	Terminal 28	Terminal 29	Terminal 30	Terminal 31	Terminal 32	Terminal 33	Terminal 34	Terminal 35	Terminal 36
Terminal 37	Terminal 38	Terminal 39	Terminal 40	Terminal 41	Terminal 42	Terminal 43	Terminal 44	Terminal 45	Terminal 46	Terminal 47	Terminal 48
Terminal 49	Terminal 50	Terminal 51	Terminal 52	Terminal 53	Terminal 54	Terminal 55	Terminal 56	Terminal 57	Terminal 58	Terminal 59	Terminal 60
Terminal 61	Terminal 62	Terminal 63	Terminal 64	Terminal 65	Terminal 66	Terminal 67	Terminal 68	Terminal 69	Terminal 70	Terminal 71	Terminal 72
Terminal 73	Terminal 74	Terminal 75	Terminal 76	Terminal 77	Terminal 78	Terminal 79	Terminal 80	Terminal 81	Terminal 82	Terminal 83	Terminal 84
Terminal 85	Terminal 86	Terminal 87	Terminal 88	Terminal 89	Terminal 90	Terminal 91	Terminal 92	Terminal 93	Terminal 94	Terminal 95	Terminal 96
Terminal 97	Terminal 98	Terminal 99	Terminal 100	Terminal 101	Terminal 102	Terminal 103	Terminal 104	Terminal 105	Terminal 106	Terminal 107	Terminal 108
Terminal 109	Terminal 110	Terminal 111	Terminal 112	Terminal 113	Terminal 114	Terminal 115	Terminal 116	Terminal 117	Terminal 118	Terminal 119	Terminal 120
Terminal 121	Terminal 122	Terminal 123	Terminal 124	Terminal 125	Terminal 126	Terminal 127	Terminal 128	Terminal 129	Terminal 130	Terminal 131	Terminal 132