


```

NN      NN      TTTTTTTTTT      000000      AAAAAA      CCCCCCCC      CCCCCCCC      FFFFFFFFFF      IIIIII      LL
NN      NN      TTTTTTTTTT      000000      AAAAAA      CCCCCCCC      CCCCCCCC      FFFFFFFFFF      IIIIII      LL
NN      NN      TT              00          00      AA          AA      CC          CC          FF          II          LL
NN      NN      TT              00          00      AA          AA      CC          CC          FF          II          LL
NNNN     NN      TT              00          00      AA          AA      CC          CC          FF          II          LL
NNNN     NN      TT              00          00      AA          AA      CC          CC          FF          II          LL
NN      NN      TT              00          00      AA          AA      CC          CC          FFFFFFFF      II          LL
NN      NN      TT              00          00      AA          AA      CC          CC          FFFFFFFF      II          LL
NN      NN      TT              0000         00      AAAAAA      AA      CC          CC          FF          II          LL
NN      NN      TT              0000         00      AAAAAA      AA      CC          CC          FF          II          LL
NN      NN      TT              00          00      AA          AA      CC          CC          FF          II          LL
NN      NN      TT              00          00      AA          AA      CC          CC          FF          II          LL
NN      NN      TT              000000        00      AA          AA      CCCCCCCC      CCCCCCCC      FF          IIIIII      LLLLLLLLLL
NN      NN      TT              000000        00      AA          AA      CCCCCCCC      CCCCCCCC      FF          IIIIII      LLLLLLLLLL

```

```

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

```


(2) 95
(3) 128
(4) 436

DECLARATIONS
NT\$EXCH_CNF - EXCHANGE DAP CONFIGURATION MESSAGES
NT\$GET_FILESPEC - BUILDS A FILESPEC


```
0000 1          $BEGIN NTOACCFIL,000,NF$NETWORK,<COMMON FILE ACCESS ROUTINES>
0000 2
0000 3
0000 4
0000 5 *****
0000 6 *
0000 7 *  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 8 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 9 *  ALL RIGHTS RESERVED.
0000 10 *
0000 11 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 12 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 13 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 14 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 15 *  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 16 *  TRANSFERRED.
0000 17 *
0000 18 *  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 19 *  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 20 *  CORPORATION.
0000 21 *
0000 22 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 23 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 24 *
0000 25 *
0000 26 *****
0000 27
0000 28
0000 29 ++
0000 30 Facility: RMS
0000 31
0000 32 Abstract:
0000 33
0000 34 This module contains commonly used file access support routines.
0000 35
0000 36 Environment: VAX/VMS, executive mode
0000 37
0000 38 Author: James A. Krycka,      Creation Date: 09-DEC-1977
0000 39
0000 40 Modified By:
0000 41
0000 42 V03-018 JEJ0039      J E Johnson      19-Jun-1984
0000 43 Fix NT$GET_FILESPEC to recognize that a resultant file
0000 44 spec has already been given back. This is of importance
0000 45 because of the possible open loop in NT$OPEN regarding to
0000 46 BRO option.
0000 47
0000 48 V03-017 JAK0145      J A Krycka      12-APR-1984
0000 49 Track changes in DAP message building algorithm.
0000 50
0000 51 V03-016 JAK0138      J A Krycka      28-MAR-1984
0000 52 Use process or system network block count value (PIO$GB_DFNBC
0000 53 or SYSS$GB_DFNBC) instead of process buffered I/O byte limit
0000 54 quota (BYTLM value from the UAF) as the basis for calculating a
0000 55 requested DAP buffer size to send to the remote FAL in the DAP
0000 56 Configuration message. SET RMS_DEFAULT and SHOW RMS_DEFAULT will
0000 57 allow the network block count value to be set from T to 127.
```


0000 58 :
0000 59 :
0000 60 :
0000 61 :
0000 62 :
0000 63 :
0000 64 :
0000 65 :
0000 66 :
0000 67 :
0000 68 :
0000 69 :
0000 70 :
0000 71 :
0000 72 :
0000 73 :
0000 74 :
0000 75 :
0000 76 :
0000 77 :
0000 78 :
0000 79 :
0000 80 :
0000 81 :
0000 82 :
0000 83 :
0000 84 :
0000 85 :
0000 86 :
0000 87 :
0000 88 :
0000 89 :
0000 90 :
0000 91 :
0000 92 :
0000 93 :--

The size of NBC affects both file transfer performance and the largest record that can be exchanged in record I/O mode. The remote FAL, of course, can force a smaller DAP buffer size (than RMS requests) to be used.

V03-015 JAK0133 J A Krycka 20-MAR-1984
Lay framework for use of logical link QIOs to transfer more than 4K bytes of data.

V03-014 RAS0223 Ron Schaefer 16-Dec-1983
Change \$SCBDEF and SCB\$xxx to \$FSCBDEF and FSCB\$xxx.

V03-013 JAK0127 J A Krycka 12-SEP-1983
Correct NT\$GET_FILESPEC to properly check for elipsis in directory component of a file specification.

V03-012 JAK0105 J A Krycka 11-MAY-1983
Enable access to large records (over 512 bytes in length) based on the negotiated DAP buffer size.

V03-011 KRM0107 K Malik 10-May-1983
Update to support DAP V7.0 specification.

V03-010 JAK0104 J A Krycka 22-APR-1983
Enhance NT\$EXCH_CNF to dynamically determine a suggested DAP buffer size to send to FAL (instead of using a predetermined value) based on BYTLM quota obtained from a call to \$GETJPI.

V03-009 KRM0079 K Malik 31-Jan-1983
Turn on indirect command file support.

V03-008 KRM0053 K Malik 10-Aug-1982
Move nodename from FWAST_NODEBUF to NWA\$T_NODEBUF (used by NT\$CRC_LOGERR)

```

0000 95      .SBTTL  DECLARATIONS
0000 96
0000 97 :
0000 98 : Include Files:
0000 99 :
0000 100
0000 101      $DAPPLGDEF      ; Define DAP prologue symbols
0000 102      $DAPHDRDEF     ; Define DAP message header
0000 103      $DAPCNFDEF     ; Define DAP Configuration message
0000 104      $DAPACCDEF     ; Define DAP Access message
0000 105      $FABDEF        ; Define File Access Block symbols
0000 106      $FWADEF        ; Define File Work Area symbols
0000 107      $IFBDEF        ; Define IFAB symbols
0000 108      $NWADEF        ; Define Network Work Area symbols
0000 109      $FSCBDEF       ; Define Scan Control Block symbols
0000 110
0000 111 :
0000 112 : Macros:
0000 113 :
0000 114 :      None
0000 115 :
0000 116 : Equated Symbols:
0000 117 :
0000 118
0000 119      ASSUME  DAP$Q_DCODE_FLG EQ 0
0000 120      ASSUME  NWA$Q_FLG EQ 0
0000 121
0000 122 :
0000 123 : Own Storage:
0000 124 :
0000 125 :      None
0000 126 :

```



```

0000 128      .SBTTL NT$EXCH_CNF - EXCHANGE DAP CONFIGURATION MESSAGES
0000 129
0000 130 :++
0000 131 : NT$EXCH_CNF - engages in a DAP dialogue to exchange DAP Configuration
0000 132 : messages with the remote FAL which includes negotiation of a DAP
0000 133 : buffer size for subsequent message exchange.
0000 134 :
0000 135 : Calling Sequence:
0000 136 :
0000 137 :     BSBW    NT$EXCH_CNF
0000 138 :
0000 139 : Input Parameters:
0000 140 :
0000 141 :     R0      Type of file access (DAP ACCFUNC code)
0000 142 :     R7      NWA address (=DAP)
0000 143 :     R8      FAB address
0000 144 :     R9      IFAB address
0000 145 :     R10     FWA address
0000 146 :     R11     Impure Area address
0000 147 :
0000 148 : Implicit Inputs:
0000 149 :
0000 150 :     DAPDEF constants
0000 151 :     DAP$W_BUF$SIZ
0000 152 :     FWA$T_NODEBUF
0000 153 :     IFB$L_DEVBUF$SIZ
0000 154 :     PIO$GB_DFNBC
0000 155 :     SYS$GB_DFNBC
0000 156 :
0000 157 : Output Parameters:
0000 158 :
0000 159 :     R0      Status code (RMS)
0000 160 :     R1-R6   Destroyed
0000 161 :     AP      Destroyed
0000 162 :
0000 163 : Implicit Outputs:
0000 164 :
0000 165 :     DAP control block
0000 166 :     NWA$W_DAPBUF$SIZ
0000 167 :     NWA$T_NODEBUF
0000 168 :     NWA$B_NODBUF$SIZ
0000 169 :     NWA$B_FILESYS
0000 170 :     NWA$B_OSTYPE
0000 171 :     NWA$Q_RCV
0000 172 :     NWA$Q_XMT
0000 173 :     NWA$Q_BIGBUF
0000 174 :     IFB$L_DEVBUF$SIZ
0000 175 :
0000 176 : Completion Codes:
0000 177 :
0000 178 :     Standard RMS completion codes
0000 179 :
0000 180 : Side Effects:
0000 181 :
0000 182 :     None
0000 183 :
0000 184 :--

```

```
0000 185
0000 186 NT$EXCH_CNF:: ; Entry point
0000 187
0000 188
0000 189 ; Obtain specified Network Block Count (NBC) value which is a sysgen parameter
0000 190 ; and also settable via the DCL command SET RMS_DEFAULT/NETWORK_BLOCK_COUNT = n.
0000 191 ;
0000 192
55 00000000'9F 98 0000 193 CVTBL @#PIO$GB_DFNBC,R5 ; Get process network block count value
0C 14 0007 194 BGTR 10$ ; Branch if specified (non-zero & pos)
55 00000000'9F 98 0009 195 CVTBL @#SYS$GB_DFNBC,R5 ; Get system network block count value
03 14 0010 196 BGTR 10$ ; Branch if specified (non-zero & pos)
55 01 D0 0012 197 MOVL #1,R5 ; We should never get here as range of
0015 198 ; DFNBC sysgen parameter is 1 to 127
7F 8F 55 91 0015 199 10$: CMPB R5,#127 ; Make internal NBC value between 1 and
02 12 0019 200 BNEQ 20$ ; 126 so that requested DAP buffer size
55 D7 001B 201 DECL R5 ; (with overhead) will be < 65536 bytes
001D 202
001D 203 ;
001D 204 ; Dispatch to file operation specific code to process network block count.
001D 205 ;
001D 206
001D 207 ASSUME DAP$K_OPEN EQ 1
001D 208 ASSUME DAP$K_CREATE EQ 2
001D 209 ASSUME DAP$K_RENAME EQ 3
001D 210 ASSUME DAP$K_ERASE EQ 4
001D 211 ASSUME DAP$K_DIR_LIST EQ 6
001D 212
001D 213 20$: $CASEB SELECTOR=R0- ; Dispatch on file access type
001D 214 BASE=#DAP$K_OPEN-
001D 215 DISPL=<-
001D 216 OPEN_CREATE- ; Open file
001D 217 OPEN_CREATE- ; Create file
001D 218 ERASE_RENAME- ; Rename file
001D 219 ERASE_RENAME- ; Erase file
001D 220 ERASE_RENAME- ; Reserved
001D 221 SEARCH- ; Search file (DAP directory list)
001D 222 > ; Any other type of access
002D 223
002D 224 ;+
002D 225 ; For erase and rename operations set NBC to one for minimum DAP buffer size.
002D 226 ;+
002D 227
002D 228 ERASE_RENAME: ; For $ERASE and $RENAME operations
55 01 D0 002D 229 MOVL #1,R5 ; One page of memory is sufficient
23 11 0030 230 BRB EXCH_COMMON ; Join common code
0032 231
0032 232 ;+
0032 233 ; For a search operation, scale down NBC (DAP buffer size) by a factor of four
0032 234 ; to reduce the time waiting for FAL to send the next buffered set of messages,
0032 235 ; especially when FAL must open each file to return file attribute information.
0032 236 ; This will help to "smooth out" the display for the DCL DIRECTORY command.
0032 237 ;
0032 238 ; However, for access to a process permanent file, set NBC to one for minimum
0032 239 ; DAP buffer size to conserve use of the process I/O segment in P1 space.
0032 240 ;+
0032 241
```



```

0032 242 SEARCH:                                     ; For $SEARCH operation
0032 243         BLBS      (R11),10$                 ; Branch if not accessing a process
0035 244         MOVL     #1,R5                     ; permanent file, else use one block
0038 245 10$:      ADDL2    #3,R5                     ; Reduce NBC to approximately one-fourth
003B 246         ASHL     #-2,R5,R5                 ; its value
0040 247         BRB      EXCH_COMMON               ; Join common code
0042 248
0042 249 ;+
0042 250 ; For open and create operations use the specified NBC value as the basis for
0042 251 ; generating requested DAP buffer size unless we have a process permanent file.
0042 252
0042 253 ; For access to a process permanent file, scale down NBC (DAP buffer size) by a
0042 254 ; factor of eight to conserve use of the process I/O segment in P1 space that
0042 255 ; is available to RMS. This reduction serves to increase the total number of
0042 256 ; process permanent files that can be simultaneously open for network access.
0042 257 ; Since the DCL OPEN command opens a process permanent file and the DCL READ and
0042 258 ; and WRITE commands are limited to 2048 byte records, the maximum NBC value
0042 259 ; will be 4 for process permanent files.
0042 260 ; -
0042 261
0042 262 OPEN_CREATE:                                   ; For $OPEN and $CREATE operations
0042 263         BLBS      (R11),EXCH_COMMON           ; Branch if not accessing a process
0045 264                                     ; permanent file
0045 265         ADDL2    #7,R5                     ; Reduce NBC to approximately one-eighth
0048 266         ASHL     #-3,R5,R5                 ; its value but not more than 4, so the
004D 267         CMPL     R0,#4                     ; resultant value is in the range of
0050 268         BLEQ     EXCH_COMMON               ; 1 to 4 blocks
0052 269         MOVL     #4,R5
0055 270
0055 271 ;+
0055 272 ; Compute DAP buffer size to request in the DAP Configuration message based on:
0055 273 ; (1) the (modified) network block count value,
0055 274 ; (2) the addition of up to 8 bytes of overhead per DAP DATA message,
0055 275 ; (3) the desire to be able to block a DAP CONTROL message with the first set
0055 276 ; of blocked DATA messages in file transfer mode,
0055 277 ; (4) the desire to have the DAP buffer fit into the nominal line buffer size
0055 278 ; of 576 bytes (which includes lower layer protocol overhead) when the NBC
0055 279 ; is one or the remote FAL can support only a one block data buffer.
0055 280 ; -
0055 281
0055 282 EXCH_COMMON:                                   ; Compute DAP buffer size to request
0055 283         CMPL     R5,#8                         ; Choose a formula based on NBC size
0058 284         BLEQ     10$                          ; to optimize requested buffer size
005A 285         MOVZWL   #<512+8>,R6                 ; Compute desired buffer size using the
005F 286         MULL2    R5,R6                       ; formula: (NBC * (512+8)) where
0062 287         BRB      EXCH_INIT                     ; NBC has a value from 9 to 126
0064 288 10$:      MOVZWL   #<512+4>,R6                 ; Compute desired buffer size using the
0069 289         MULL2    R5,R6                       ; formula: (NBC * (512+4) + 28) where
006C 290         ADDL2    #28,R6                     ; NBC has a value from 1 to 8
006F 291
006F 292 ;+
006F 293 ; Initialize the DAP control block and the transmit and receive buffers in the
006F 294 ; NWA. These buffers will be used to exchange DAP Configuration messages, then
006F 295 ; they may be replaced by larger DAP buffers if the negotiated DAP buffer size
006F 296 ; is larger than NWA$C_BUFFERSIZ. Note that the transmit buffer is used for
006F 297 ; both building a new DAP message (BLD descriptor) and for concatenating DAP
006F 298 ; messages before sending them to FAL (XMT descriptor).
```

```
006F 299 :-
006F 300
006F 301 EXCH_INIT:                                     ; Initialize control block and buffers
006F 302 $ZERO_FILL-                                   ; Zero DAP control block
006F 303 DST=(R7)-
006F 304 SIZE=#DAP$C_BLN
0077 305
0077 306 ASSUME NWA$C_BUFFERSIZ GE <512+4+28>
0077 307 ASSUME NWA$Q_XMT EQ NWA$Q_RCV+8
0077 308
50 00E0 C7 7E 0077 309 MOVAQ NWA$Q_RCV(R7),R0          ; Get start address of descriptors
80 01A0 C7 9E 007C 310 CLRL (R0)+                    ; Initialize receive descriptor
80 03C0 C7 9E 007E 311 MOVAB NWA$T_RCVBUF(R7),(R0)+
0220 8F B0 0083 312 CLRL (R0)+                    ; Initialize transmit descriptor
00CA C7 B0 0085 313 MOVAB NWA$T_XMTBUF(R7),(R0)+
008A 314 MOVW NWA$C_BUFFERSIZ,-
008E 315 NWA$W_DAPBUFSIZ(R7)                        ; Make the preallocated buffer size
0091 316 ;+                                           ; the current DAP buffer size
0091 317 ;+
0091 318 ; Build and send DAP Configuration message to partner.
0091 319 :-
0091 320
0091 321 SEND_CNF:                                     ; (required message)
0091 322 $SETBIT #NWA$V_LAST_MSG,(R7)                ; Declare this last message to block
50 01 D0 0095 323 MOVL #DAP$K_CNF_MSG,R0             ; Get message type value
FF65' 30 0098 324 BSBW NT$BUICD_HEAD                 ; Construct message header
85 56 B0 009B 325 MOVW R6,(R5)+                     ; Store BUFSIZ field (desired value)
85 07 90 009E 326 MOVW #DAP$K_VAXVMS,(R5)+           ; Store OSTYPE field
85 03 90 00A1 327 MOVW #DAP$K_RMS32,(R5)+            ; Store FILESYS field
85 07 90 00A4 328 MOVW #DAP$K_VERNUM_V,(R5)+         ; Store VERNUM field
85 00 90 00A7 329 MOVW #DAP$K_ECONUM_V,(R5)+         ; Store ECONUM field
85 00 90 00AA 330 MOVW #DAP$K_USRNUM_V,(R5)+         ; Store USRNUM field
85 04 90 00AD 331 MOVW #DAP$K_DECVER_V,(R5)+         ; Store DECVER field
85 00 90 00B0 332 MOVW #DAP$K_USRVER_V,(R5)+         ; Store USRVER field
51 EFF67DF7 8F D0 00B3 333 MOVL #DAP$K_SYSCAPT_V,R1   ; Get VAX supported capabilities
52 00001962 8F D0 00BA 334 MOVL #DAP$K_SYSCAPT2_V,R2  ; quadword mask
FF3C' 30 00C1 335 BSBW NT$CVT_BN8_EXT               ; Store SYSCAP as an extensible field
FF39' 30 00C4 336 BSBW NT$BUICD_TAIL                 ; Finish building message
FF36' 30 00C7 337 BSBW NT$TRANSMIT                  ; Send Configuration message to FAL
03 50 E8 00CA 338 BLBS R0,RCV_CNF                    ; Branch on success
009A 31 00CD 339 ERROR1: BRW ERROR                     ; Branch on failure
00D0 340
00D0 341 ;+
00D0 342 ; Receive DAP Configuration message response from partner.
00D0 343 :-
00D0 344
00D0 345 RECV_CNF:                                     ; (required message)
00D0 346 $SETBIT #DAP$K_CNF_MSG,DAP$L_MSG_MASK(R7)    ; Expect response of configuration msg
00D5 347 BSBW NT$RECEIVE                               ; Get reply from FAL
FF28' 30 00D5 348 BLBC R0,ERROR1                     ; Branch on failure
F2 50 E9 00D8 349 MOVW DAP$B_OSTYPE(R7),-           ; Save OSTYPE field in NWA
42 A7 90 00DB 350 MOVW NWA$B_OSTYPE(R7),-
00C4 C7 00DE 351 DAP$B_FILESYS(R7),-
43 A7 90 00E1 352 MOVW NWA$B_FILESYS(R7),-           ; Save FILESYS field in NWA
00C5 C7 00E4 353 NWA$B_FILESYS(R7)
00E7 354
00E7 355 ;+
```



```
00E7 356 : Determine the 'agreed upon' DAP buffer size to use and save this value.
00E7 357 : It is the smaller of partner's buffer size and our requested buffer size.
00E7 358 :-
00E7 359
00CA C7 56 B0 00E7 360 MOVW R6,NWASW_DAPBUFSIZ(R7) : Assume we'll use requested buffer size
      40 A7 B5 00EC 361 TSTW DAPSW_BUFSIZ(R7) : Use our buffer size if partner
      0C 13 00EF 362 BEQL 10$ : has unlimited buffer space
56 40 A7 B1 00F1 363 CMPW DAPSW_BUFSIZ(R7),R6 : Use our buffer size if partner
      06 1E 00F5 364 BGEQU 10$ : has buffer size GEQ ours
      40 A7 B0 00F7 365 MOVW DAPSW_BUFSIZ(R7),- : Use partner's buffer size which is
00CA C7 00FA 366 NWASW_DAPBUFSIZ(R7) : smaller than ours
      00FD 367
      00FD 368 :+
      00FD 369 : Allocate big DAP buffers if standard size buffers already allocated as part of
      00FD 370 : the NWA are not sufficient. Note that the transmit buffer will be twice the
      00FD 371 : size of the receive buffer (or twice NWASW_DAPBUFSIZ) as it is used for both
      00FD 372 : building new DAP messages and for concatenating DAP messages before sending
      00FD 373 : them to FAL. The overflow buffer space may be used when a new message is being
      00FD 374 : constructed and there are messages already blocked in the transmit buffer.
      00FD 375 :-
      00FD 376
56 00CA C7 3C 00FD 377 10$: MOVZWL NWASW_DAPBUFSIZ(R7),R6 : Get negotiated DAP buffer size
0220 8F 56 B1 0102 378 CMPW R6,#NWASW_BUFFERSIZ : Use standard buffers if they are large
      22 1B 0107 379 BLEQU 20$ : enough
      56 07 C0 0109 380 ADDL2 #7,R6 : Round up buffer size to quadword
      56 07 CA 010C 381 BICL2 #7,R6 : boundary
52 56 03 C5 010F 382 MULL3 #3,R6,R2 : Compute size of desired receive buffer
      0113 383 : plus a double-length transmit buffer
      0113 384 BSBW RMSGETPAG : Allocate space (NOT ZEROED)
      51 50 E9 0116 385 BLBC R0,ERROR : Branch on failure
0170 C7 52 7D 0119 386 MOVQ R2,NWASQ_BIGBUF(R7) : Update big buffer descriptor
00E4 C7 53 D0 011E 387 MOVL R3,NWASQ_RCV+4(R7) : Update receive descriptor
      53 56 C0 0123 388 ADDL2 R6,R3 : Move pointer to next buffer
00EC C7 53 D0 0126 389 MOVL R3,NWASQ_XMT+4(R7) : Update transmit descriptor
      012B 390
      012B 391 :+
      012B 392 : Determine the maximum record size that can be supported for network access
      012B 393 : in record I/O operations. This is accomplished by examining the negotiated
      012B 394 : DAP buffer size and then updating the device buffer size value in the IFAB
      012B 395 : (if appropriate from its initial setting in NT$MOD_DEV_CHAR).
      012B 396
      012B 397 : The value in IFBSL_DEVBUFFSIZ establishes the network record size limit as
      012B 398 : this value is used by RMS at $CONNECT time to allocate the BDB buffer. The
      012B 399 : size of this buffer determines the largest record that can be moved to/from
      012B 400 : user's buffer during $GET, $PUT, and $UPDATE operations on a remote file.
      012B 401
      012B 402 : The algorithm establishes a maximum record size that is equal to 1 to 64 pages
      012B 403 : of memory (i.e., 512, 1024, ..., 32768 bytes).
      012B 404
      012B 405 : Note that IFBSL_DEVBUFFSIZ does not limit the size of a user block I/O request
      012B 406 : which can be from 1 to 127 blocks.
      012B 407 :-
      012B 408
56 00CA C7 3C 012B 409 20$: MOVZWL NWASW_DAPBUFSIZ(R7),R6 : Get negotiated DAP buffer size
      56 08 C2 0130 410 SUBL2 #8,R6 : Subtract DAP DATA message overhead
56 56 F7 8F 78 0133 411 ASHL #-9,R6,R6 : Compute # whole pages
      0F 13 0138 412 BEQL FINISH : Keep initial value if DAPBUFSIZ < 520
```

```

      40 8F 56 91 013A 413      CMPB R6,#64      ; Limit value to 64 pages as the largest
      56 40 04 1B 013E 414      BLEQU 30$      ; record defined by RMS is slightly
48 A9 56 8F 9A 0140 415      MOVZBL #64,R6      ; less than 32K bytes
      56 09 78 0144 416 30$: ASHL #9,R6,IFB$;DEVBUFSIZ(R9);Compute largest record size supported
      0149 417
      0149 418
      0149 419 ;+
      0149 420 ; While we have both a FWA and a NWA, move the nodename (sans delimiters or
      0149 421 ; access strings) and the nodename size to NWA$T_NODEBUF & NWA$B_NODBUFSIZ
      0149 422 ; for use by NT$CRC_LOGERR.
      0149 423 ;-
      0149 424
      0149 425 FINISH:
      53 07E9 CA 9E 0149 426      MOVAB FWA$T_NODEBUF(R10),R3 ; Miscellaneous
      63 07 22 3A 014E 427      LOCC #^A/'7,#FWA$C_MAXNODNAM+1 ; Get address of nodename (spec list)
      04 12 0152 428      BNEQ 10$ ; (R3) ; Search for quote
      63 07 3A 3A 0154 429      LOCC #^A/;/,#FWA$C_MAXNODNAM+1 ; Branch if access control string
      52 51 53 C3 0158 430 10$: SUBL3 R3,R1,R2 ; (R3) ; Find the colon (must be there)
0169 C7 63 52 28 015C 431      MOVCL R3,R1,R2 ; Compute the nodename length
      0168 C7 52 90 0162 432      MOVCL R2,(R3),NWA$T_NODEBUF(R7);Move nodename to NWA
      05 016A 433      MOVCL R2,NWA$B_NODBUFSIZ(R7) ; Move length to NWA
      434 ERROR: RMSSUC ; Return success
      RSB ; Exit with RMS code in R0
```



```
016B 436 .SBTTL NT$GET_FILESPEC - BUILDS A FILESPEC
016B 437
016B 438 :++
016B 439 : NT$GET_FILESPEC - builds a filespec (less primary node name) from its
016B 440 : constituent parts and stores it as a counted ASCII string.
016B 441 :
016B 442 : Calling Sequence:
016B 443 :
016B 444 :     BSBW    NT$GET_FILESPEC
016B 445 :
016B 446 : Input Parameters:
016B 447 :
016B 448 :     R5      Address of buffer to receive counted ASCII string
016B 449 :     R7      NWA address
016B 450 :     R8      FAB address
016B 451 :     R9      IFAB address
016B 452 :     R10     FWA address
016B 453 :     R11     Impure Area address
016B 454 :
016B 455 : Implicit Inputs:
016B 456 :
016B 457 :     FWASB_DIRTERM
016B 458 :     FWASQ_DEVICE
016B 459 :     FWASQ_DIR1
016B 460 :     FWASQ_DIR2
016B 461 :     FWASQ_DIR2+8 thru FWASQ_DIR2+48
016B 462 :     FWASQ_NAME
016B 463 :     FWASQ_QUOTED
016B 464 :     FWASQ_VERSION
016B 465 :     FWASV_DEVICE
016B 466 :     FWASV_EXP_VER
016B 467 :     FWASV_GRPMBR
016B 468 :     FWASV_DIR
016B 469 :     FWASV_DIR_LVL
016B 470 :     FWASV_QUOTED
016B 471 :     NWA$B_OSType
016B 472 :
016B 473 : Output Parameters:
016B 474 :
016B 475 :     R0-R3   Destroyed
016B 476 :     R5      Updated buffer pointer (address of end of string + i)
016B 477 :     AP      Destroyed
016B 478 :
016B 479 : Implicit Outputs:
016B 480 :
016B 481 :     None
016B 482 :
016B 483 : Completion Codes:
016B 484 :
016B 485 :     None
016B 486 :
016B 487 : Side Effects:
016B 488 :
016B 489 :     None
016B 490 :
016B 491 : --
016B 492
```

```
0150 8F BB 016B 493 NT$GET_FILESPEC::      ; Entry point
53 55 DO 016B 494      PUSHR    #^M<R4,R6,R8>    ; Save registers
      83 94 016F 495      MOVL    R5,R3          ; Copy next byte pointer
58 53 DO 0172 496      CLRB     (R3)+          ; Skip over count byte
      53 DO 0174 497      MOVL    R3,R8          ; Save pointer to start of DST string
      0177 498
      0177 499
      0177 500      ; Process secondary node spec strings.
      0177 501
      0177 502
      2F AA 95 0177 503      TSTB     FWASB_SUBNODCNT(R10) ; Branch if there is only one
      24 13 017A 504      BEQL     10$          ; node spec in node spec list
51 50 01B4 CA 3C 017C 505      MOVZWL  FWASQ_NODE1(R10),R0 ; Get size of primary node spec
52 00D8 CA 50 A3 0181 506      SUBW3   R0,FWASQ_NODE(R10),R1 ; Compute descriptor of concatenated
00DC CA 50 C1 0187 507      ADDL3   R0,FWASQ_NODE+4(R10),R2 ; secondary node spec strings
63 62 51 28 018D 508      MOVC3   R1,(R2),R3) ; Copy secondary node specs
      0191 509
      0191 510
      0191 511      ; Process quoted string.
      0191 512
      0191 513      Note: If there is only a primary node spec, then the quoted string is copied
      0191 514      with the quote delimiters removed. Conversely, if secondary node specs
      0191 515      are present, then the quoted string is copied with the quote delimiters
      0191 516      intact.
      0191 517
      0191 518
      04 6A 35 E0 0191 519      BBS     #FWASV_REMRESULT,(R10),5$ ; Branch if result already delivered.
      22 6A 1A E1 0195 520      BBC     #FWASV_QUOTED,(R10),30$ ; Branch if no quoted string follows
50 0170 CA 7D 0199 521 5$:      MOVQ    FWASQ_QUOTED(R10),R0 ; Get descriptor of quoted string
      019E 522      ; (including quote delimiters)
      019E 523      BRB     20$          ; Join common code
      F5 6A 35 E0 01A0 524 10$:      BBS     #FWASV_REMRESULT,(R10),5$ ; Branch if result already delivered.
      13 6A 1A E1 01A4 525      BBC     #FWASV_QUOTED,(R10),30$ ; Branch if no quoted string follows
50 0170 CA 02 C3 01A8 526      SUBL3   #2,FWASQ_QUOTED(R10),R0 ; Get size of string less quotes
51 0174 CA 01 C1 01AE 527      ADDL3   #1,FWASQ_QUOTED+4(R10),R1 ; Get address of string
63 61 50 28 01B4 528 20$:      MOVC3   R0,(R1),R3) ; Copy quoted string
      009B 31 01B8 529      BRW     120$        ; Join common code
      01BB 530
      01BB 531
      01BB 532      ; Process device name.
      01BB 533
      01BB 534
      0B 6A 0F E1 01BB 535 30$:      BBC     #FWASV_DEVICE,(R10),40$ ; Branch if no device name present
      00E0 CA 28 01BF 536      MOVC3   FWASQ_DEVICE(R10),- ; Copy device name
63 00E4 DA 90 01C3 537      @FWASQ_DEVICE+4(R10),(R3)
      83 3A 90 01C7 538      MOVB     #^A\:\,(R3)+ ; Append delimiter
      01CA 539
      01CA 540
      01CA 541      ; Process directory list.
      01CA 542      ; It is either in the [group,member] or [directory_name_list] format.
      01CA 543
      01CA 544
      4A 6A 0E E1 01CA 545 40$:      BBC     #FWASV_DIR,(R10),90$ ; Branch if no directory present
      0A AA 02 83 01CE 546      SUBB3   #2,FWASB_DIRTERM(R10),- ; Store left bracket ('[' or '<')
      83 01D2 547      (R3)+ ; (ASCII code is right bracket + 2)
      2A 6A 1B E0 01D3 548      BBS     #FWASV_GRPMBR,(R10),70$ ; Branch if [group,member] format
56 0130 CA 7E 01D7 549      MOVAQ   FWASQ_DIR1(R10),R6 ; Get address of directory descriptor
```



```

      1D EF 01DC 550 EXTZV #FWASV_DIR_LVL$,-      : Get # of directory sub-levels
SC 6A 03 01DE 551 #FWASV_DIR_LVL$(R10),AP;      : (0 means UFD level only)
      50 86 D0 01E1 552 50$: MOVL (R6)+,R0      : Get size of string
63 96 50 28 01E4 553 MOVC3 R0,@(R6)+,(R3)      : Copy next (sub)directory name
      83 2E 90 01E8 554 MOVB #^A\.,(R3)+      : Copy directory separator
05 F8 A6 10 E1 01EB 555 BBC #FSCBSV_ELIPS,-8(R6),60$ : Branch if elipsis does not follow
      8F B0 01F0 556 : (sub)directory name
      8E 5C F4 01F5 557 MOVW #^A\.,.,(R3)+      : Append two dots to make an elipsis
17 F8 A6 10 E0 01F8 558 60$: SOBGEQ AP,50$      : Branch if more directory names
      53 D7 01FD 559 BBS #FSCBSV_ELIPS,-8(R6),80$ : Branch if we just copied an elipsi
      13 11 01FF 560 DECL R3                  : Otherwise, remove unwanted (single)
63 0130 CA 28 0201 561 BRB 80$                : trailing dot
      0134 DA 90 0205 562 70$: MOVC3 FWASQ_DIR1(R10),-      : Copy group directory field
      83 2C 90 0209 563 @FWASQ_DIR1+4(R10),(R3)      :
      0138 CA 28 020C 564 MOVB #^A\.,(R3)+      : Copy directory separator
63 013C DA 28 0210 565 MOVC3 FWASQ_DIR2(R10),-      : Copy member directory field
      83 0A AA 90 0214 566 @FWASQ_DIR2+4(R10),(R3)      :
      567 80$: MOVB FWASB_DIRTERM(R10),(R3)+; Store right bracket ('J' or '>')
      568
      569 :
      570 : Process file name, file type, and file version.
      571 : To facilitate communication with non-VMS systems, several system specific
      572 : version number checks will be made.
      573 :
      574 : Note: The file name string described by FWASQ_NAME is guaranteed to contain
      575 : both the "." and ";" delimiters, even if the user did not specify a
      576 : file type or file version number. Furthermore, a "." version number
      577 : delimiter entered by the user will have been converted to a ";"
      578 : delimiter by RMOXPFN!
      579 :
      580 :
63 0170 CA 28 0218 581 90$: MOVC3 FWASQ_NAME(R10),-      : Copy file name string (assembled
      0174 DA 90 021C 582 @FWASQ_NAME+4(R10),(R3)      : into one string by RMOXPFN)
32 67 34 E0 0220 583 BBS #DAP$V_VAXVMS,(R7),120$ : Branch if partner is VAX/VMS
      584
      585 :
      586 : If the remote node is not VMS, delete the trailing semi-colon (null version
      587 : number) if the user did not explicitly enter a version # in the primary
      588 : filespec string.
      589 :
      590 :
      591 BBS #FWASV_EXP_VER,(R10),100$;Branch if version # was explicit
OE 6A 10 E0 0224 592 CMPB -1(R3),#^A\;      : Is last character a semi-colon?
3B FF A3 91 0228 593 BNEQ 100$              : Branch if not
      08 12 022C 594 DECL R3                  : Otherwise delete it here and from
      53 D7 022E 595 DECW FWASQ_NAME(R10)      : filename descriptor in FWA
      0170 CA B7 0230 596 BRB 120$            : All done
      20 11 0234 597
      598 :
      599 : If the remote node is RT-11, remove the version number substring (either
      600 : ";;" or ";;ver") because RT-11 does not recognize the version number element.
      601 :
      602 :
50 0180 CA 3C 0236 603 100$: MOVZWL FWASQ_VERSION(R10),R0 : Get number of digits in version #
      50 D6 023B 604 INCL R0                  : Add semi-colon delimiter to count
      OA 67 38 E1 023D 605 BBS #DAP$V_RT11,(R7),110$ : Branch if remote node is not RT-11
      53 50 C2 0241 606 SUBL2 R0,R3          : Delete version number substring here
```



```
0170 CA 50 A2 0244 607 SUBW2 R0,FWA$Q_NAME(R10) ; and from filename descriptor in FWA
        0B 11 0249 608 BRB 120$ ; All done
        024B 609
        024B 610 ;
        024B 611 ; If the remote node is TOPS-20, convert the ":" version number delimiter to a
        024B 612 ; ":" delimiter because TOPS-20 requires uses the semi-colon character as a
        024B 613 ; file attribute delimiter.
        024B 614 ;
        024B 615 ;
        07 67 37 E1 024B 616 110$: BBC #DAP$V TOPS20,(R7),120$ ; Branch if remote node is not TOPS-20
51 53 50 C3 024F 617 SUBL3 R0,R3,R1 ; Calculate address of delimiter
    61 2E 93 0253 618 MOV B #^A\.\,(R1) ; Convert period to semi-colon
        0256 619
        0256 620 ;
        0256 621 ; Finish building counted ASCII string.
        0256 622 ;
        0256 623 ;
        50 53 58 C3 0256 624 120$: SUBL3 R8,R3,R0 ; Calculate size of string
        FF A8 50 90 025A 625 MOV B R0,-1(R8) ; Store the count
        55 53 D0 025E 626 MOVL R3,R5 ; Put next byte pointer in proper reg
        0150 8F BA 0261 627 POP R #^M<R4,R6,R8> ; Restore registers
        05 0265 628 RSB ; Exit
        0266 629
        0266 630 .END ; End of module
```


\$\$PSECT_EP = 00000000
\$\$COUNT = 00000006
\$\$RMSTEST = 0000001A
\$\$RMS_PBUGCHK = 00000010
\$\$RMS_TBUGCHK = 00000008
\$\$RMS_UMODE = 00000004
DAP\$B_ACCFUNC 00000040
DAP\$B_ACCOPT 00000041
DAP\$B_BITCNT 00000035
DAP\$B_DCODE_FID 00000019
DAP\$B_DCODE_MAC 0000001B
DAP\$B_DCODE_MSG 0000001A
DAP\$B_DECVER 00000047
DAP\$B_ECONUM 00000045
DAP\$B_FAC 00000042
DAP\$B_FILESYS 00000043
DAP\$B_FLAGS 00000031
DAP\$B_LEN256 00000034
DAP\$B_LENGTH 00000033
DAP\$B_OSTYPE 00000042
DAP\$B_SHR 00000043
DAP\$B_STREAMID 00000032
DAP\$B_TYPE 00000030
DAP\$B_USRNUM 00000046
DAP\$B_USRVER 00000048
DAP\$B_VERNUM 00000044
DAP\$B_X_FIELD 00000024
DAP\$C_BLN 000000C0
DAP\$K_BLN 000000C0
DAP\$K_CNF_MSG = 00000001
DAP\$K_CREATE = 00000002
DAP\$K_DECVER_V = 00000004
DAP\$K_DIR_LIST = 00000006
DAP\$K_ECONUM_V = 00000000
DAP\$K_ERASE = 00000004
DAP\$K_OPEN = 00000001
DAP\$K_RENAME = 00000003
DAP\$K_RMS32 = 00000003
DAP\$K_SYSCAP1_V = EFF67DF7
DAP\$K_SYSCAP2_V = 00001962
DAP\$K_USRNUM_V = 00000000
DAP\$K_USRVER_V = 00000000
DAP\$K_VAXVMS = 00000007
DAP\$K_VERNUM_V = 00000007
DAP\$L_CHWA 00000030
DAP\$L_CRC_RSLT 00000020
DAP\$L_DCODE_STS 00000018
DAP\$L_MSG_MASK 0000001C
DAP\$L_SSPQA 00000080
DAP\$L_TEMP 00000090
DAP\$M_BITCNT = 00000008
DAP\$M_DSP_3NAM = 00000200
DAP\$M_GET = 00000002
DAP\$M_GO_NOGO = 00000010
DAP\$M_MSE = 00000010
DAP\$M_SEGMENT = 00000040
DAP\$M_TMP1\$ = 000000C0

DAP\$M_TMP2\$ = 0000FC00
DAP\$Q_DCODE_FLG 00000000
DAP\$Q_FILESPEC 00000044
DAP\$Q_MSG_BUF1 00000008
DAP\$Q_MSG_BUF2 00000010
DAP\$Q_PASSWORD 00000050
DAP\$Q_SYSCAP 00000028
DAP\$Q_SYSPEC 00000038
DAP\$V_RT11 = 00000038
DAP\$V_TOPS20 = 00000037
DAP\$V_VAXVMS = 00000034
DAP\$W_BUFSIZ 00000040
DAP\$W_DISPLAY1 0000004C
DAP\$W_PARTNER 00000006
DAP\$W_VERSION 00000004
ERASE_RENAME 0000002D R 01
ERROR 0000016A R 01
ERROR1 000000CD R 01
EXCH_COMMON 00000055 R 01
EXCH_INIT 0000006F R 01
FINISH 00000149 R 01
FSCBSV_ELIPS = 00000010
FWASB_DIRTERM = 0000000A
FWASB_SUBNODCNT = 0000002F
FWASC_MAXNODNAM = 00000006
FWASQ_DEVICE = 000000E0
FWASQ_DIR1 = 00000130
FWASQ_DIR2 = 00000138
FWASQ_NAME = 00000170
FWASQ_NODE = 000000D8
FWASQ_NODE1 = 000001B4
FWASQ_QUOTED = 00000170
FWASQ_VERSION = 00000180
FWASS_DIR_LVL5 = 00000003
FWAST_NODEBUF = 000007E9
FWASV_DEVICE = 0000000F
FWASV_DIR = 0000000E
FWASV_DIR_LVL5 = 0000001D
FWASV_EXP_VER = 00000010
FWASV_GRPMBR = 0000001B
FWASV_QUOTED = 0000001A
FWASV_REMRESULT = 00000035
IFBSL_DEVBUFSIZ = 00000048
NT\$BUILD_HEAD ***** X 01
NT\$BUILD_TAIL ***** X 01
NT\$CVT_BN8_EXT ***** X 01
NT\$EXCH_CNF 00000000 RG 01
NT\$GET_FILESPEC 0000016B RG 01
NT\$RECEIVE ***** X 01
NT\$TRANSMIT ***** X 01
NWB\$B_ALLXABCNT 0000011C
NWB\$B_DAP_RAC 000000C9
NWB\$B_FILESYS 000000C5
NWB\$B_KEYXABCNT 0000011D
NWB\$B_NETSTRSIZ 0000016F
NWB\$B_NODBUFSIZ 00000168
NWB\$B_ORG 000000C6

NTOACCFIL
Symbol table

COMMON FILE ACCESS ROUTINES

D 8

15-SEP-1984 23:48:14 VAX/VMS Macro V04-00
5-SEP-1984 16:20:11 [RMS.SRC]NTOACCFIL.MAR;1

Page 15
(4)

NWASB_OSTYPE 000000C4
NWASB_RFM 000000C7
NWASB_RMS_RAC 000000C8
NWASC_BLN = 00000800
NWASC_BUFFERSIZ 00000220
NWASK_BLN 00000800
NWASL_ALLXABADR 00000100
NWASL_DATXABADR 00000104
NWASL_DEV 000000C0
NWASL_FHCXABADR 00000108
NWASL_KEYXABADR 0000010C
NWASL_MSG_MASK 000000D4
NWASL_PROXABADR 00000110
NWASL_RDTXABADR 00000114
NWASL_SAVE_FLGS 00000128
NWASL_SUMXABADR 00000118
NWASL_THREAD 000000FC
NWASL_XLTATTR 00000238
NWASL_XLTBUFFLG 0000022C
NWASL_XLTCNT 00000228
NWASL_XLTMAXINDX 00000234
NWASL_XLTSIZ 00000230
NWASQ_ACS 00000244
NWASQ_BIGBUF 00000170
NWASQ_BLD 000000F0
NWASQ_FLG 00000000
NWASQ_INODE 0000025C
NWASQ_IOSB 000000D8
NWASQ_LNODE 00000160
NWASQ_LOGNAME 0000023C
NWASQ_NCB 00000264
NWASQ_RCV 000000E0
NWASQ_SAVE_DESC 00000120
NWASQ_XLTBUF1 0000024C
NWASQ_XLTBUF2 00000254
NWASQ_XMT 000000E8
NWASt_ACSBUF 0000026C
NWASt_AUXBUF 000005E0
NWASt_DAP 00000000
NWASt_INODEBUF 000004AC
NWASt_ITM_ATTR 00000200
NWASt_ITM_END 00000224
NWASt_ITM_LST 00000200
NWASt_ITM_MAXINDX 00000218
NWASt_ITM_STRING 0000020C
NWASt_NCBBUF 0000052C
NWASt_NODEBUF 00000169
NWASt_RCVBUF 000001A0
NWASt_SCAN 00000100
NWASt_TEMP 00000120
NWASt_XLTBUF1 000002AC
NWASt_XLTBUF2 000003AC
NWASt_XMTBUF 000003C0
NWASV_LAST_MSG = 00000000
NWASW_BUILD 000000D2
NWASW_DAPBUFSIZ 000000CA
NWASW_DIR_OFF 000000CC

NWASW_DISPLAY 000000D0
NWASW_FIL_OFF 000000CE
NWASW_JNLXABJOP 0000011E
OPEN_CREATE 00000042 R 01
PIO\$GB_DFNBC ***** X 01
RECV_CNF 000000D0 R 01
RMSGETPAG ***** X 01
SEARCH 00000032 R 01
SEND_CNF 00000091 R 01
SYS\$GB_DFNBC ***** X 01

000000D0
000000CE
0000011E
00000042 R 01
***** X 01
000000D0 R 01
***** X 01
00000032 R 01
00000091 R 01
***** X 01

+-----+
! Psect synopsis !
+-----+

PSECT name	Allocation	PSECT No.	Attributes															
ABS	00000000 (0.)	00 (0.)	NOPIC	USR	CON	ABS	LCL	NOSHR	NOEXE	NORD	NOWRT	NOVEC	BYTE					
NF\$NETWORK	00000266 (614.)	01 (1.)	PIC	USR	CON	REL	GBL	NOSHR	EXE	RD	NOWRT	NOVEC	BYTE					
\$AB\$\$	00000800 (2048.)	02 (2.)	NOPIC	USR	CON	ABS	LCL	NOSHR	EXE	RD	WRT	NOVEC	BYTE					

+-----+
! Performance indicators !
+-----+

Phase	Page faults	CPU Time	Elapsed Time
Initialization	30	00:00:00.06	00:00:00.63
Command processing	141	00:00:00.62	00:00:04.20
Pass 1	342	00:00:12.65	00:00:38.01
Symbol table sort	0	00:00:01.72	00:00:02.40
Pass 2	124	00:00:02.65	00:00:06.58
Symbol table output	22	00:00:00.16	00:00:00.33
Psect synopsis output	2	00:00:00.03	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	663	00:00:17.90	00:00:52.19

The working set limit was 1500 pages.
67627 bytes (133 pages) of virtual memory were used to buffer the intermediate code.
There were 70 pages of symbol table space allocated to hold 1186 non-local and 24 local symbols.
630 source lines were read in Pass 1, producing 14 object records in Pass 2.
24 pages of virtual memory were used to define 23 macros.

+-----+
! Macro library statistics !
+-----+

Macro library name	Macros defined
-\$255\$DUA28:[RMS.OBJ]RMS.MLB;1	15
-\$255\$DUA28:[SYSLIB]STARLET.MLB;2	4
TOTALS (all libraries)	19

1400 GETS were required to define 19 macros.
There were no errors, warnings or information messages.
MACRO/LIS=LIS\$:NTOACCFIL/OBJ=OBJ\$:NTOACCFIL MSRC\$:NTOACCFIL/UPDATE=(ENH\$:NTOACCFIL)+LIB\$:RMS/LIB

0315 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

NT0ACCESS
LIS

NT0CLOSE
LIS

NT0BLDXAB
LIS

NT0CONN
LIS

NT0CREATE
LIS

NT0DAP10
LIS

NT0DAPCRC
LIS

NT0ACCFIL
LIS

NT0BLK10
LIS