

RRRRRRRRRRRR		MMM		MMM	SSSSSSSSSSSS
RRRRRRRRRRRR		MMM		MMM	SSSSSSSSSSSS
RRRRRRRRRRRR		MMM		MMM	SSSSSSSSSSSS
RRR	RRR	MMMMMM	MMMMMM	SSS	
RRR	RRR	MMMMMM	MMMMMM	SSS	
RRR	RRR	MMMMMM	MMMMMM	SSS	
RRR	RRR	MMM	MMM	SSS	
RRR	RRR	MMM	MMM	SSS	
RRR	RRR	MMM	MMM	SSS	
RRRRRRRRRRRR		MMM		SSSSSSSSSS	
RRRRRRRRRRRR		MMM		SSSSSSSSSS	
RRRRRRRRRRRR		MMM		SSSSSSSSSS	
RRR	RRR	MMM			SSS
RRR	RRR	MMM			SSS
RRR	RRR	MMM			SSS
RRR	RRR	MMM			SSS
RRR	RRR	MMM			SSS
RRR	RRR	MMM			SSS
RRR	RRR	MMM			SSS
RRR	RRR	MMM		SSSSSSSSSSSS	
RRR	RRR	MMM		SSSSSSSSSSSS	
RRR	RRR	MMM		SSSSSSSSSSSS	

SY

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

NT

PI

```
RRRRRRRR  MM      MM      SSSSSSSS  000000  MM      MM      AAAAAA  GGGGGGGG  TTTTTTTTTT  AAAAAA
RRRRRRRR  MM      MM      SSSSSSSS  000000  MM      MM      AAAAAA  GGGGGGGG  TTTTTTTTTT  AAAAAA
RR      RR  MMMM  MMMM  SS      00      00  MMMM  MMMM  AA      AA  GG      TT      AA      AA
RR      RR  MMMM  MMMM  SS      00      00  MMMM  MMMM  AA      AA  GG      TT      AA      AA
RR      RR  MM      MM      SS      00      0000  MM      MM      AA      AA  GG      TT      AA      AA
RR      RR  MM      MM      SS      00      0000  MM      MM      AA      AA  GG      TT      AA      AA
RRRRRRRR  MM      MM      SSSSSS  00      00      00  MM      MM      AA      AA  GG      TT      AA      AA
RRRRRRRR  MM      MM      SSSSSS  00      00      00  MM      MM      AA      AA  GG      TT      AA      AA
RR      RR  MM      MM      SS      0000      00  MM      MM      AAAAAAAAAA  GG      GGGGGG  TT      AAAAAAAAAA
RR      RR  MM      MM      SS      0000      00  MM      MM      AAAAAAAAAA  GG      GGGGGG  TT      AAAAAAAAAA
RR      RR  MM      MM      SS      00      00  MM      MM      AA      AA  GG      TT      AA      AA
RR      RR  MM      MM      SS      00      00  MM      MM      AA      AA  GG      TT      AA      AA
RR      RR  MM      MM      SSSSSSSS  000000  MM      MM      AA      AA  GG      TT      AA      AA
RR      RR  MM      MM      SSSSSSSS  000000  MM      MM      AA      AA  GG      TT      AA      AA
                                     ....
                                     ....
                                     ....
                                     ....
```

```
LL      IIIIII  SSSSSSSS
LL      IIIIII  SSSSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SSSSSS
LL      II      SSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LLLLLLLLLL  IIIIII  SSSSSSSS
LLLLLLLLLL  IIIIII  SSSSSSSS
```


(2)	93
(3)	121
(9)	334
(10)	372
(11)	414
(14)	495

DECLARATIONS
\$SPACE ROUTINE
\$NXTVOL ROUTINE
RMSREWIND MT - INTERNAL ROUTINE TO REWIND MT
MTFUNC - SUBROUTINE TO ISSUE A MAGTAPE CONTROL QIO
RMSWTAPMARK WRITE TAPE MARKS


```
0000 1          $BEGIN RMSOMAGTA,000,RMSRMS,<$NXTVOL, $SPACE & OTHER MAGTAPE CODE>
0000 2
0000 3
0000 4 *****
0000 5 *
0000 6 *   COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 7 *   DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 8 *   ALL RIGHTS RESERVED.
0000 9 *
0000 10 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 11 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 12 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 13 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 14 *  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 15 *  TRANSFERRED.
0000 16 *
0000 17 *  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 18 *  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 19 *  CORPORATION.
0000 20 *
0000 21 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 22 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 23 *
0000 24 *
0000 25 *****
0000 26
0000 27 ++
0000 28 Facility: RMS32
0000 29
0000 30 Abstract:
0000 31 This module provides the $SPACE and $NXTVOL RMS services as well
0000 32 as the internal magtape rewind file code.
0000 33
0000 34 Environment:
0000 35 STAR processor running STARLET exec.
0000 36
0000 37 Author: L F Laverdure,          creation date: 13-DEC-1977
0000 38
0000 39 Modified By:
0000 40
0000 41 V03-003 JEJ0026          J E Johnson          11-Apr-1984
0000 42 Tie off invalid network operations.
0000 43
0000 44 V03-002 DGB0008          Donald G. Blair      01-Mar-1984
0000 45 Change the way the ACP is called as part of the
0000 46 restructuring necessary for access mode protected files.
0000 47
0000 48 V03-001 KBT0173          Keith B. Thompson    23-Aug-1982
0000 49 Reorganize psects
0000 50
0000 51 V02-020 CDS0001          C Saether            20-Dec-1981
0000 52 Clear BIO_LAST after rabset.
0000 53
0000 54 V02-019 DMW0001          David Michael Walp    30-Nov-1981
0000 55 Return EOF error rather than DPE if "END OF VOLUME" error
0000 56 and foreign tape
0000 57
```



```
0000 58 : V2-018 MCN0001 Maria del C. Nasr 25-Aug-1980
0000 59 : After the NXTVOL function is completed, if not at EOF, the
0000 60 : position context should be destroyed, by clearing the
0000 61 : offset into the block, setting NRP_VBN to 1, and EBK to -1.
0000 62 :
0000 63 : V02-017 REFORMAT Keith B. Thompson 29-Jul-1980
0000 64 :
0000 65 : V016 PSK0012 P S Knibbe 15-Feb-1980 01:20
0000 66 : on space eof context should be destroyed. IRB$V_EOF bit
0000 67 : is cleared, NRP_VBN is set to one and ebk is set to -1
0000 68 :
0000 69 : V015 PSK0011 P S Knibbe 13-Dec-1979 03:40
0000 70 : if an eof is returned on an ansi magtape, set the IRB$V_EOF bit
0000 71 : and set IRB$V_NRP_VBN equal to ebk to make sure that the bit
0000 72 : gets tested.
0000 73 :
0000 74 : V014 PSK0001 P S Knibbe 20-Nov-1979 06:30
0000 75 : added routine to write tape marks at end of file.
0000 76 : also changed rewind to clear IFB$V_EOF bit and IRB$V_EOF bit
0000 77 :
0000 78 : V013 JAK0020 J A Krycka 05-Sep-1979 12:00
0000 79 : release 2.0 work.
0000 80 :
0000 81 : V011 RAN0004 R A Newell 20-Dec-1978 10:15
0000 82 : eliminate wait on tape rewind i/o function.
0000 83 :
0000 84 : Revision History:
0000 85 :
0000 86 : L F Laverdure, 31-Oct-1978 10:40 : set nowait for foreign mt rewind
0000 87 : R A Newell, 18-Sep-1978 09:09 : changed entry pt name for isam fit
0000 88 : R A Newell, 31-Aug-1978 12:01 : eliminate wait for i/o on mbx
0000 89 :
0000 90 :
0000 91 :
```

```
0000 93      .SBTTL  DECLARATIONS
0000 94
0000 95 :
0000 96 : Include Files:
0000 97 :
0000 98
0000 99 :
0000 100 : Macros:
0000 101 :
0000 102
0000 103      $$$DEF
0000 104      $FIBDEF
0000 105      $RABDEF
0000 106      $DEVDEF
0000 107      $IFBDEF
0000 108      $IRBDEF
0000 109      $IODEF
0000 110      $RMSDEF
0000 111
0000 112 :
0000 113 : Equated Symbols:
0000 114 :
0000 115
0000 116 :
0000 117 : Own Storage:
0000 118 :
0000 119
```



```
0000 121      .SBTTL $SPACE ROUTINE
0000 122
0000 123 :++
0000 124
0000 125      RMS$SPACE  -
0000 126
0000 127      this routine spaces a magtape or disk file forward or backward
0000 128      a given number of blocks.  the file must be accessed for block i/o.
0000 129
0000 130      Calling sequence:
0000 131
0000 132      entered as a result of user's calling sys$space
0000 133
0000 134      Input Parameters:
0000 135
0000 136      AP      user's argument list
0000 137
0000 138      Implicit Inputs:
0000 139
0000 140      the contents of the rab, in particular, the bkt field gives the number
0000 141      of blocks to space, forward if positive, backward if negative.
0000 142
0000 143      Output Parameters:
0000 144
0000 145      R0      status code
0000 146      R1      destroyed
0000 147
0000 148      Implicit Outputs:
0000 149
0000 150      the sts field of the rab
0000 151      the stv field of the rab (set to # of blocks actually spaced, always
0000 152      positive)
0000 153
0000 154      Completion Codes:
0000 155
0000 156      standard rms
0000 157
0000 158      Side Effects:
0000 159
0000 160      none
0000 161
0000 162 :--
0000 163
0000 164
0000 165 :++
0000 166
0000 167      verify operation allowed (accessed for block i/o to a magtape or disk)
0000 168      and call internal space magtape routine unless this is for a disk file.
0000 169
0000 170 :--
0000 171
0000 172      $ENTRY RMS$SPACE
0000 173      $STPT  SPACE
0000 174      $RABSET ANYFAC=1,-      ; set up stream
0000 175      BIO=1,-                ; any fac value o.k., must be accessed
0000 176      CFLG=1                 ; for block i/o, clear find last
000A 177
```

```
31 6A 3E E0 000A 178 CSB #IRBSV_BIO_LAST,(R9) ; Clear bio_last.
03 6A 0E E0 000E 179 BBS #IFBSV_DAP,(R10),NTSPACE ; branch if network operation
      00A0 31 0016 181 BRW #DEVSV_FOD,IFBSL_PRIM_DEV(R10),10$
      FFE4' 30 0019 182 10$: BSBW RMS$FLUSH ; write all dirty blocks
      21 50 E9 001C 183 BLBC R0,EXIT ; get out on error
      20 A9 D4 001F 184 CLRL IRBSL_CURBDB(R9) ; say no current bdb
      44 A9 B4 0022 185 CLRW IRBSW_NRP_OFF(R9) ; start at beginning next block
56 38 A8 D0 0025 186 MOVL RABSL_BKT(R8),R6 ; pick up space count
      05 E1 0029 187 BBC #DEVSV_SQD,- ; branch if not magtape
      3C 6A 002B 188 IFBSL_PRIM_DEV(R10),SPDISK
      002D 189 CSB #IRBSV_EOF,(R9) ; clear the eof bit - we're no
      0031 190 ; longer at eof
40 A9 01 D0 0031 191 MOVL #1,IRBSL_NRP_VBN(R9) ; nrp should equal 1 to avoid
      0035 192 ; problems comparing with ebk
74 AA 01 CE 0035 193 MNEGL #1,IFBSL_EBK(R10) ; ebk must be -1 for same reason
      0085 30 0039 194 BSBW RM$SPACE_MT ; space the magtape
0C A8 56 D0 003C 195 SPEXIT: MOVL R6,RABSL_STV(R8) ; set stv from block count
      FFBD' 31 0040 196 EXIT: BRW RM$EXRMS ; back to user
      0043 197
      0043 198 ;++
      0043 199 ;
      0043 200 ; perform network space function
      0043 201 ;
      0043 202 ;--
      0043 203
      0043 204 NTSPACE:
51 38 A8 D0 0043 205 MOVL RABSL_BKT(R8),R1 ; get # blocks to space
      FFBD' 30 0047 206 BSBW NT$SPACE ; space the file via remote fal
82D2 8F 50 B1 004A 207 CMPW R0,#<RMSS_SUP&^XFFFF> ; screen out unsupported error
      EF 13 004F 208 BEQL EXIT ; as rab$stv already has
      0051 209 ; associated dap error code
C144 8F 50 B1 0051 210 CMPW R0,#<RMSS_SUPPORT&^XFFFF> ; and let us not forget the other form
      E8 13 0056 211 BEQL EXIT ; of the unsupported dap error code
      38 A8 D5 0058 212 TSTL RABSL_BKT(R8) ; space forward request?
      06 19 005B 213 BLSS 10$ ; branch if not
40 A9 56 C0 005D 214 ADDL2 R6,IRBSL_NRP_VBN(R9) ; update next block pointer
      D9 11 0061 215 BRB SPEXIT ; join mainline
      0063 216
40 A9 56 C2 0063 217 10$: SUBL2 R6,IRBSL_NRP_VBN(R9) ; update next block pointer
      D7 11 0067 218 EXIT ; exit to user
      0069 219
```



```
0069 221
0069 222 :
0069 223 : $space for disk file.
0069 224 :
0069 225 : update nbp checking for beginning and end of file conditions
0069 226 :
0069 227 :
0069 228 SPDISK:
40 A9 2F 19 0069 229 BLSS 20$ : branch if backspace
56 C0 006B 230 ADDL2 R6,IRB$L_NRP_VBN(R9) : bump nrp by space count
74 AA C3 006F 231 SUBL3 IFB$L_EBK(R10),- : past eof?
51 40 A9 0072 232 IRB$L_NRP_VBN(R9),R1
01 3F 1B 0075 233 BLEQU 30$ : branch if not
51 D1 0077 234 CMPL R1,#1 : exactly 1 block past?
05 12 007A 235 BNEQ 10$ : branch if not
5C AA B5 007C 236 TSTW IFB$W_FFB(R10) : something in last block?
35 12 007F 237 BNEQ 30$ : branch if yes (o.k.)
0081 238 :
0081 239 :
0081 240 : attempt to move past eof block (+1 if ffb not = 0).
0081 241 : position to eof block (+1 if ffb not = 0) and return an error.
0081 242 :
0081 243 :
40 A9 74 AA D0 0081 244 10$: MOVL IFB$L_EBK(R10),IRB$L_NRP_VBN(R9); set to eof block
5C AA B5 0086 245 TSTW IFB$W_FFB(R10) : block empty?
05 13 0089 246 BEQL 15$ : branch if yes
40 A9 D6 008B 247 INCL IRB$L_NRP_VBN(R9) : bump past partial blk
51 D7 008E 248 DECL R1 : adjust difference count
56 51 C2 0090 249 15$: SUBL2 R1,R6 : adjust space count
0093 250 : = (# wanted) - (# lacking)
0093 251 RMSERR EOF ; indicate not all spaced
1C 11 0098 252 BRB 30$ ; continue
009A 253 :
009A 254 :
009A 255 : space backward for disk file.
009A 256 :
009A 257 : check for beginning of file.
009A 258 :
009A 259 :
40 56 56 CE 009A 260 20$: MNEGL R6,R6 : get # blks to backspace
A9 56 D1 009D 261 CMPL R6,IRB$L_NRP_VBN(R9) : can we do entire backspace?
OF 1F 00A1 262 BLSSU 25$ : branch if yes
00A3 263 RMSERR BOF : no - change status code
56 40 A9 D0 00A8 264 MOVL IRB$L_NRP_VBN(R9),R6 : & adjust actual space count
18 E0 00AC 265 BBS #DEV$V_FOR,- : branch if mounted foreign
02 6A 00AE 266 IFB$L_PRIM_DEV(R10),25$
56 D7 00B0 267 DECL R6 : leave nbp = 1
40 A9 56 C2 00B2 268 25$: SUBL2 R6,IRB$L_NRP_VBN(R9) : adjust nbp by space count
FF83 31 00B6 269 30$: BRW SPEXIT : go finish up
00B9 270
```

```

00B9 272 :++
00B9 273 :
00B9 274 : handle invalid device error.
00B9 275 :
00B9 276 :--
00B9 277 :
00B9 278 ERRIOP:
00B9 279 RMSERR IOP
FF3F' 31 00BE 280 BRW RM$EXRMS ; get out

```



```
00C1 282
00C1 283 :++
00C1 284 :
00C1 285 RM$SPACE MT - space file blocks routine for magtape.
00C1 286 RM$WRITEOF -
00C1 287 :
00C1 288 inputs:
00C1 289 :
00C1 290 R6 # of blocks to space (forward if postive, backward if negative)
00C1 291 R8 rab/fab address
00C1 292 R9 irab/ifab address
00C1 293 R10 ifab address
00C1 294 R11 impure area address
00C1 295 :
00C1 296 outputs:
00C1 297 :
00C1 298 R0 status code
00C1 299 R6 # blocks actually spaced
00C1 300 R1-R5,AP destroyed
00C1 301 :
00C1 302 :--
00C1 303 :
00C1 304 RM$SPACE MT::
00C1 305 BBS #DEV$V FOR,- ; branch if mounted foreign
00C3 306 IFB$L PRIM DEV(R10),20$ ;
55 0A 6A E0 00C5 307 MOVZBL #FIB$C_SPACE,R5 ; magtape control function code
73 10 00C8 308 BSBB MTFNC1 ; do the space
56 0E A9 3C 00CA 309 10$: MOVZWL IRB$L_IOS+2(R9),R6 ; get # of blocks spaced
05 00CE 310 RSB ; and return to caller
00CF 311 :
00CF 312 :
00CF 313 : do space logical blocks for non-files-11 magtape
00CF 314 :
00CF 315 :
50 26 3C 00CF 316 20$: MOVZWL #IOS$ SKIPRECORD,R0 ; lgogical space function code
5A 10 00D2 317 BSBB MTFUNC_FOR ; do the space
F4 11 00D4 318 BRB 10$
00D6 319 :
00D6 320 :++
00D6 321 :
00D6 322 : write an eof mark for close of output foreign magtape (or mailbox)
00D6 323 :
00D6 324 :--
00D6 325 :
50 28 3C 00D6 326 RM$WRITEOF::
14 E1 00D9 327 MOVZWL #IOS$ WRITEOF,R0
04 69 00DB 328 BBC #DEV$V MBX,- ; branch if not mailbox
50 40 8F 88 00DD 329 IFB$L PRIM DEV(R9),NOTMBX
00E1 330 BISB2 #IOSM_NOW,R0 ; eliminate wait for i/o
49 11 00E1 331 NOTMBX:
332 BRB MTFUNC_FOR1
```

```
00E3 334      .SBTTL $NXTVOL ROUTINE
00E3 335
00E3 336      :++
00E3 337      :
00E3 338      Entry point for $nxtvol.
00E3 339      :
00E3 340      check that this is for a magtape and, if so, do a $flush followed by
00E3 341      call to the magtape primitive to do the next volume processing.
00E3 342      :
00E3 343      :--
00E3 344
00E3 345      $ENTRY  RM$NXTVOL
00E3 346      $TSTPT  NXTVOL
00E9 347      $RABSET      ; set up stream
3E   E0 00ED 348      BBS      #IFB$V DAP,-      ; branch if a network device
6A   00EF 349      IFB$L PRIM_DEV(R10),-
21   00F0 350      NTNXTVOL
05   E1 00F1 351      BBC      #DEV$V SQD,-      ; branch if not magtape
6A   00F3 352      IFB$L PRIM_DEV(R10),-
C4   00F4 353      ERRIOP
FF08' 30 00F5 354      BSBW      RM$FLUSH      ; write any dirty blocks
14 50  E9 00F8 355      BLBC      R0,10$      ; get out on error
55   03 9A 00FB 356      MOVZBL  #FIB$C_NEXTVOL,R5      ; set magtape control function code
3B   10 00FE 357      BSBB      MTFUNC      ; do the nextvol call
0B 69 21  E0 0100 358      BBS      #IRB$V EOF,(R9),10$      ; if at EOF, return
44 A9 D4 0104 359      CLRL      IRB$L_NRP_OFF(R9)      ; set record offset to zero
40 A9 01 D0 0107 360      MOVL      #1,IRB$L_NRP_VBN(R9)      ; reset VBN to 1
74 AA 01 CE 010B 361      MNEGL      #1,IFB$L_EBK(R10)      ; indicate beginning of file section
FEEE' 31 010F 362 10$:      BRW      RM$EXRMS      ; back to user
0112 363
0112 364      :
0112 365      : Do the nextvol for a network device.
0112 366      :
0112 367
0112 368      NTNXTVOL:
FEEB' 30 0112 369      BSBW      NT$NXTVOL      ; Do the next vol. operation
FEEB' 31 0115 370      BRW      RM$EXRMS      ; and return to the user.
```



```
0118 372 .SBTTL RMSREWIND_MT - INTERNAL ROUTINE TO REWIND MT
0118 373
0118 374 :++
0118 375 :
0118 376 : RMSREWIND_MT -
0118 377 :
0118 378 : rewind magtape to beginning of file
0118 379 :
0118 380 :
0118 381 : inputs:
0118 382 :
0118 383 : R8 rab/fab address
0118 384 : R9 irab/ifab address
0118 385 : R10 ifab address
0118 386 : R11 impure area address
0118 387 :
0118 388 : outputs:
0118 389 :
0118 390 : R0 status code
0118 391 : R1-R6,AP destroyed
0118 392 :
0118 393 :--
0118 394
0118 395 RMSREWIND_MT::
0118 396 MOVZWL #FIB$C_REWINDFIL,R5 ; set mt control function code
0118 397 BBC #DEV$V_FOR,- ; branch if not mounted foreign
0118 398 IFB$C_PRIM_DEV(R10),MTFUNC
0118 399
0118 400 MOVZWL #IOS$REWIND!IOSM_NOWAIT,R0 ; rewind tape i/o function
0124 401 CSB #IFB$V_EOF,(R10) ; no longer at end of file
0128 402 CSB #IRB$V_EOF,(R9) ; clear bit in irab also
012C 403 MTFUNC_FOR1:
012C 404 CLRL R6 ; zero qio p1 parameter
012E 405
012E 406 MTFUNC_FOR:
012E 407 CLRQ -(SP) ; p6 = p5 = 0 for qio
0130 408 CLRQ -(SP) ; p4 = p3 = 0
0132 409 CLRL -(SP) ; p2 = 0
0134 410 PUSHL R6 ; p1
0136 411 BSBW RM$FCPFNC_NOFIB ; perform the qio
0139 412 BRB MTSTS
```

55 06 3C 0118 396
18 E1 0118 397
1C 6A 011D 398
50 00A4 8F 3C 011F 399
0124 401
0128 402
56 D4 012C 403
012C 404
012E 405
012E 406
7E 7C 012E 407
7E 7C 0130 408
7E D4 0132 409
56 DD 0134 410
FEC7' 30 0136 411
2B 11 0139 412


```
0138 414 .SBTTL MTFUNC - SUBROUTINE TO ISSUE A MAGTAPE CONTROL QIO
0138 415
0138 416 :++
0138 417 : mtfunc subroutine to issue user magtape control functions to the
0138 418 : magtape primitive.
0138 419 :
0138 420 : entry at mtfnc1 if r6 contains a magtape control value.
0138 421 :
0138 422 : inputs:
0138 423 :
0138 424 : R5 magtape control function code
0138 425 : R6 magtape control value (not an input for entry at mtfunc)
0138 426 : R8 rab/fab address
0138 427 : R9 irab/ifab address
0138 428 : R10 ifab address
0138 429 : R11 impure area address
0138 430 :
0138 431 : outputs:
0138 432 :
0138 433 : R0 status code
0138 434 : R6 zeroed if entry at mtfunc, else unchanged
0138 435 : R1-R5,AP destroyed
0138 436 :
0138 437 : note: RM$FLUSH must already have been called for file
0138 438 :
0138 439 :--
0138 440
52 56 D4 0138 441 MTFUNC: CLRL R6 ; zero control value param
0138 442 MTFNC1: ; entry point with r6 = control value
0138 443 MOVZBL #FIB$C_LENGTH,R2 ; set size of fib
0138 444 BSBW RM$GETSPC1 ; allocate fib
0138 445 BLBC R0,MTXIT ; branch on error
0138 446
0138 447 :
0138 448 : build descriptor for fib on stack
0138 449 :
0138 450
7E 51 DD 0147 451 PUSHL R1 ; addr of fib
16 40 8F 9A 0149 452 MOVZBL #FIB$C_LENGTH,-(SP) ; length of fib
18 A1 55 B0 014D 453 MOVW R5,FIB$W_CNTRLFUNC(R1) ; set control function code
50 56 D0 0151 454 MOVL R6,FIB$L_CNTRLVAL(R1) ; and value
0138 455 MOVZWL #IOS_ACPCONTROL,R0 ; set i/o function code
0138 456 CLRQ -(SP) ; p6 = p5 = 0 (qio arguments)
0138 457 BSBW RM$FCPFNC P4 ; call acp and perform function
0138 458 POPR #^M<R2,R4> ; clean stack & get fib len & addr
0138 459 PUSHL R0 ; save status code
0138 460 BSBW RM$RETSPC1 ; deallocate fib
0138 461 POPR #^M<R0> ; restore status
01 50 E9 0166 462 MTSTS: BLBC R0,MTERR ; branch on error
05 0169 463 MTXIT: RSB ; return to caller
```



```
016A 465
016A 466 :
016A 467 : Had an error. Map to RMS version of status code.
016A 468 :
016A 469 :
0870 8F 50 B1 016A 470 MTERR: CMPW R0,#SS$_ENDOFFILE ; did we get end of file error?
                                BEQL 10$ ; branch if not
09A0 8F 50 B1 016F 471 ;
                                CMPW R0,#SS$_ENDOFVOLUME ; did we have end of volume
                                BNEQ 40$ ; branch if not
                                BBC #DEV$V FOR,- ; and foreign mount
                                IFB$$_PRIM_DEV(R10),50$ ; branch if not
                                BRB 20$ ;
                                BBS #DEV$V FOR,- ; if not foreign
                                IFB$$_PRIM_DEV(R10),30$ ;
                                #IRB$$_EOF,(R9) ; set the eof flag
40 A9 74 AA D0 0182 479 20$: SSB #IRB$$_EOF,(R9) ;
44 A9 5C AA B0 0186 480 MOVW IFB$$_EBK(R10),IRB$$_NRP_VBN(R9) ;
                                MOVW IFB$$_FFB(R10),IRB$$_NRP_OFF(R9) ; make sure the bit gets tested
                                RMSERR EOF ; yes - change to rms version
                                RSB
                                05 0195 483
                                0196 484
0938 8F 50 B1 0196 485 40$: CMPW R0,#SS$_BEGOFFILE ; was it beginning of file?
                                BNEQ 50$ ; branch if not
                                RMSERR BOF ; yes - change to rms version
                                RSB
                                05 01A2 488
                                01A3 489
                                01A3 490 50$:
                                01A3 491 ; default error code
FE55' 31 01A8 492 BRW RMS$MAPERR ; map error to rms & return
                                01AB 493
```



```
01AB 495 .SBTTL RMSWTTAPMARK WRITE TAPE MARKS
01AB 496
01AB 497 :++
01AB 498
01AB 499 RMSWTTAPMARK -
01AB 500
01AB 501
01AB 502 this routine writes tape marks on foreign magtapes if
01AB 503 the IFBSV_EOF bit is set. this bit gets set whenever a put
01AB 504 is done and cleared on rewind.
01AB 505
01AB 506 the tape marks do not get written if the file was not
01AB 507 opened with write access.
01AB 508
01AB 509 input
01AB 510
01AB 511 none
01AB 512
01AB 513 implicit input
01AB 514
01AB 515 R9 - ifab address
01AB 516 IFBSV_WRTACC - set if writing is permitted
01AB 517 IFBSV_EOF - set if at end of file
01AB 518
01AB 519 output
01AB 520
01AB 521 R0 - set to appropriate status code
01AB 522
01AB 523 implicit output
01AB 524
01AB 525 none
01AB 526
01AB 527 side effects
01AB 528
01AB 529 none
01AB 530
01AB 531 :--
01AB 532
01AB 533 RMSWTTAPMARK::
1D 69 30 E1 01AB 534 BBC #IFBSV_WRTACC,(R9),40$ ; write accessed?
19 69 21 E1 01AF 535 BBC #IFBSV_EOF,(R9),40$ ; eof ?
FF20 30 01B3 536 BSBW RMSWRITEOF ; yes - write eof
16 50 E9 01B6 537 BLBC R0,30$ ; write another one
FF1A 30 01B9 538 BSBW RMSWRITEOF
10 50 E9 01BC 539 BLBC R0,30$
01BF 540
56 01 CE 01BF 541 MNEGL #1,R6 ; space some
FEFC 30 01C2 542 BSBW RM$SPACE_MT
01C5 543
827A 8F 50 B1 01C5 544 CMPW R0,#RMSS_EOF&^XFFFF ; make sure error is eof
03 12 01CA 545 BNEQ 30$ ; if not - real error
01CC 546
01CC 547 40$: RMSSUC
05 01CF 548 30$: RSB
01D0 549 .END
```


RMSOMAGTA
Symbol table

\$NXTVOL, \$SPACE & OTHER MAGTAPE CODE ^{F 4}

16-SEP-1984 01:22:12 VAX/VMS Macro V04-00
5-SEP-1984 16:25:05 [RMS.SRC]RMSOMAGTA.MAR;1

Page 14
(14)

```

$$PSECT_EP      = 00000000
$$RMSTEST       = 0000001A
$$RMS_PBUGCHK   = 00000010
$$RMS_TBUGCHK   = 00000008
$$RMS_UMODE     = 00000004
DEVSV_FOD       = 0000000E
DEVSV_FOR       = 00000018
DEVSV_MBX       = 00000014
DEVSV_SQD       = 00000005
ERRIOP         = 000000B9 R      01
EXIT           = 00000040 R      01
FIBSC_LENGTH    = 00000040
FIBSC_NEXTVOL   = 00000003
FIBSC_REWINDFIL = 00000006
FIBSC_SPACE     = 00000004
FIBSL_CNTRLVAL  = 00000018
FIBSW_CNTRLFUNC = 00000016
IFBSL_EBK       = 00000074
IFBSL_PRIM_DEV  = 00000000
IFBSV_DAP       = 0000003E
IFBSV_EOF       = 00000021
IFBSV_WRTACC    = 00000030
IFBSW_FFB       = 0000005C
IOSM_NOW        = 00000040
IOSM_NOWAIT     = 00000080
IOS_ACPCONTROL  = 00000038
IOS_REWIND      = 00000024
IOS_SKIPRECORD  = 00000026
IOS_WRITEOF     = 00000028
IRBSL_CURBDB    = 00000020
IRBSL_IOS       = 0000000C
IRBSL_NRP_OFF   = 00000044
IRBSL_NRP_VBN   = 00000040
IRBSV_BIO_LAST  = 00000027
IRBSV_EOF       = 00000021
IRBSW_NRP_OFF   = 00000044
MTERR          = 0000016A R      01
MTFNC1         = 0000013D R      01
MTFUNC         = 0000013B R      01
MTFUNC_FOR     = 0000012E R      01
MTFUNC_FOR1    = 0000012C R      01
MTSTS         = 00000166 R      01
MTXIT         = 00000169 R      01
NOTMBX        = 000000E1 R      01
NT$NXTVOL      = ***** X      01
NT$SPACE       = ***** X      01
NTNXTVOL       = 00000112 R      01
NTSPACE        = 00000043 R      01
PIOSA_TRACE    = ***** X      01
RABSL_BKT      = 00000038
RABSL_STV      = 0000000C
RMSEXAMS       = ***** X      01
RMSFCPFNC_NOF1B = ***** X      01
RMSFCPFNC_P4   = ***** X      01
RMSFLUSH       = ***** X      01
RMSGETSPC1     = ***** X      01
RMSMAPERR      = ***** X      01

```

```

RMSRETSPC1      ***** X      01
RMSREWIND_MT    00000118 RG      01
RMSRSET         ***** X      01
RMSSPACE_MT     000000C1 RG      01
RMSWRITEOF      000000D6 RG      01
RMSWTTAPMARK    000001AB RG      01
RMS$NXTVOL      = 000000E1 RG      01
RMSSSPACE       = FFFFFFFE RG      01
RMSS_BOF        = 00018198
RMSS_DPE        = 0001C03A
RMSS_EOF        = 0001827A
RMSS_IOP        = 00018574
RMSS_SUP        = 000182D2
RMSS_SUPPORT     = 0001C144
SPDISK         = 00000069 R      01
SPEXIT         = 0000003C R      01
SS$_BEGOFFILE   = 000000938
SS$_ENDOFFILE   = 000000870
SS$_ENDOFVOLUME = 0000009A0
TPT$NXTVOL      ***** X      01
TPT$SPACE       ***** X      01

```

RMS
V04

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

PSECT name	Allocation	PSECT No.	Attributes
-----	-----	-----	-----
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR
RMSRMS	000001D0 (464.)	01 (1.)	PIC USR
\$AB\$\$	00000000 (0.)	02 (2.)	NOPIC USR

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

Phase	Page faults	CPU Time	Elapsed Time
-----	-----	-----	-----
Initialization	36	00:00:00.07	00:00:01.74
Command processing	137	00:00:00.83	00:00:07.20
Pass 1	413	00:00:15.45	00:00:35.81
Symbol table sort	0	00:00:02.52	00:00:03.54
Pass 2	101	00:00:02.96	00:00:05.47
Symbol table output	10	00:00:00.12	00:00:00.22
Psect synopsis output	2	00:00:00.03	00:00:00.03
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	701	00:00:21.99	00:00:54.02

The working set limit was 1650 pages.
88641 bytes (174 pages) of virtual memory were used to buffer the intermediate code.
There were 90 pages of symbol table space allocated to hold 1765 non-local and 22 local symbols.
549 source lines were read in Pass 1, producing 14 object records in Pass 2.
25 pages of virtual memory were used to define 24 macros.

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

Macro library name	Macros defined
-----	-----
_\$255\$DUA28:[RMS.OBJ]RMS.MLB;1	13
_\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	0
_\$255\$DUA28:[SYSLIB]STARLET.MLB;2	7
TOTALS (all libraries)	20

1885 GETs were required to define 20 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS\$:RMSOMAGTA/OBJ=OBJ\$:RMSOMAGTA MSRC\$:RMSOMAGTA/UPDATE=(ENH\$:RMSOMAGTA)+EXECML\$/LIB+LIB\$:RMS/LIB

0330 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

