```
RRRRRRRRRRR      MMM          MMM     SSSSSSSSSSS
RRRRRRRRRRRR     MMM          MMM     SSSSSSSSSSS
RRRRRRRRRRRR     MMM          MMM     SSSSSSSSSSS
RRR         RRR  MMMMMM    MMMMMM  SSS
RRR         RRR  MMMMMM    MMMMMM  SSS
RRR         RRR  MMMMMM    MMMMMM  SSS
RRR         RRR  MMM   MMM   MMM   SSS
RRR         RRR  MMM   MMM   MMM   SSS
RRR         RRR  MMM   MMM   MMM   SSS
RRRRRRRRRRR      MMM          MMM     SSSSSSSSS
RRRRRRRRRRRR     MMM          MMM     SSSSSSSSS
RRRRRRRRRRRR     MMM          MMM     SSSSSSSSS
RRR   RRR        MMM          MMM             SSS
RRR   RRR        MMM          MMM             SSS
RRR   RRR        MMM          MMM             SSS
RRR      RRR     MMM          MMM             SSS
RRR      RRR     MMM          MMM             SSS
RRR      RRR     MMM          MMM             SSS
RRR         RRR  MMM          MMM  SSSSSSSSSSS
RRR         RRR  MMM          MMM  SSSSSSSSSSS
RRR         RRR  MMM          MMM  SSSSSSSSSSS
```

FILEID**RMOEXTEND

```
RRRRRRRR    MM      MM    000000    EEEEEEEEEE  XX      XX  TTTTTTTTTT  EEEEEEEEEE  NN      NN  DDDDDDDD
RRRRRRRR    MM      MM    000000    EEEEEEEEEE  XX      XX  TTTTTTTTTT  EEEEEEEEEE  NN      NN  DDDDDDDD
RR      RR  MMMM  MMMM  00      00  EE            XX  XX      TT      EE          NN      NN  DD      DD
RR      RR  MMMM  MMMM  00      00  EE            XX  XX      TT      EE          NN      NN  DD      DD
RR      RR  MM  MM  MM  00    0000  EE              XX        TT      EE          NNNN    NN  DD      DD
RR      RR  MM  MM  MM  00    0000  EE              XX        TT      EE          NNNN    NN  DD      DD
RRRRRRRR    MM      MM  00  00  00  EEEEEEEE        XX        TT      EEEEEEEE    NN  NN  NN  DD      DD
RRRRRRRR    MM      MM  00  00  00  EEEEEEEE        XX        TT      EEEEEEEE    NN  NN  NN  DD      DD
RR  RR      MM      MM  0000    00  EE            XX  XX      TT      EE          NN    NNNN  DD      DD
RR  RR      MM      MM  0000    00  EE            XX  XX      TT      EE          NN    NNNN  DD      DD
RR    RR    MM      MM  00      00  EE          XX      XX    TT      EE          NN      NN  DD      DD
RR    RR    MM      MM  00      00  EE          XX      XX    TT      EE          NN      NN  DD      DD
RR      RR  MM      MM    000000    EEEEEEEEEE  XX      XX    TT      EEEEEEEEEE  NN      NN  DDDDDDDD
RR      RR  MM      MM    000000    EEEEEEEEEE  XX      XX    TT      EEEEEEEEEE  NN      NN  DDDDDDDD

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
```

Table of contents

(2) 116 DECLARATIONS
(3) 149 RM$EXTEND0 - COMMON FILE EXTEND ROUTINE
(9) 390 RM$JNL_EXTEND - Journal extend operations

```
0000     1          $BEGIN  RMOEXTEND,000,RMS$RMS0,<COMMON EXTEND FILE ROUTINE>
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 ;                   Routine to perform common file extend processing for
0000    32 ;                   all file organizations.
0000    33 ;
0000    34 ; ENVIRONMENT:
0000    35 ;                   STAR processor running STARLET EXEC.
0000    36 ;
0000    37 ; AUTHOR: L F Laverdure,          CREATION DATE: 2-Dec-1977
0000    38 ;
0000    39 ; MODIFIED BY:
0000    40 ;
0000    41 ;       V03-015 RAS0284          Ron Schaefer            29-Mar-1984
0000    42 ;               Fix error paths to put the area id in the STV.
0000    43 ;
0000    44 ;       V03-014 DGB0015          Donald G. Blair         02-Mar-1984
0000    45 ;               Allocate full-length FIB to support access mode protected
0000    46 ;               files.
0000    47 ;
0000    48 ;       V03-013 KPL0009          Peter Lieberwirth       25-Oct-1983
0000    49 ;               Automatic relative extends on $PUT are broken because
0000    50 ;               relative code puts ALQ in R6.  JNL_EXTEND thinks a non-zero
0000    51 ;               R6 contains an XAB address.  Fix by forcing relative file
0000    52 ;               automatic extend on $PUT journaling to just journal ALQ,
0000    53 ;               as if no XAB is ever present.  RMSREC will thereby use
0000    54 ;               the ALQ as used by the extension logic, which is fine.
0000    55 ;
0000    56 ;       V03-012 DAS0001          David Solomon           16-Sep-1983
0000    57 ;               Journal actual final ALQ (after extend).
```

```
0000     58 ;
0000     59 ;       V03-011 KPL0008         Peter Lieberwirth       27-Jul-1983
0000     60 ;               Add more information to the EXTEND RJR entry - include
0000     61 ;               fields specified in the XABALL if there is an XABALL.
0000     62 ;               Journal EXTENDs of all file organizations.
0000     63 ;
0000     64 ;       V03-010 KPL0007         Peter Lieberwirth        7-Jun-1983
0000     65 ;               Fix error path on journal write after successful extend.
0000     66 ;
0000     67 ;       V03-008 KPL0006         Peter Lieberwirth       31-May-1983
0000     68 ;               Fill JNL type in extend MJB.
0000     69 ;
0000     70 ;       V03-007 KPL0005         Peter Lieberwirth       26-May-1983
0000     71 ;               Support new RJR format.
0000     72 ;
0000     73 ;       V03-006 KPL0004         Peter Lieberwirth        1-May-1983
0000     74 ;               Add omitted definitions.
0000     75 ;
0000     76 ;       V03-005 KPL0003         Peter Lieberwirth        1-May-1983
0000     77 ;               Fix branch out of range and typo.
0000     78 ;
0000     79 ;       V03-004 KPL0002         Peter Lieberwirth       30-Apr-1983
0000     80 ;               Oops!  Don't journal the extend unless we're journaling.
0000     81 ;
0000     82 ;       V03-004 KPL0001         Peter Lieberwirth       30-Apr-1983
0000     83 ;               After-image journal ISAM EXTENDs.
0000     84 ;
0000     85 ;       V03-003 KBT0331         Keith B. Thompson       10-Sep-1982
0000     86 ;               Remove $FRBDEF
0000     87 ;
0000     88 ;       V03-002 KBT0205         Keith B. Thompson       23-Aug-1982
0000     89 ;               Reorganize psects
0000     90 ;
0000     91 ;       V03-001 KBT0119         Keith B. Thompson        6-Aug-1982
0000     92 ;               Remove the ref. to set_sifb_adr
0000     93 ;
0000     94 ;       V02-012 JWH0001         Jeffrey W. Horn 2-Mar-1982
0000     95 ;               Get rid of hack put in durring reformat that left
0000     96 ;               a check for invalid ALN options inoperative.
0000     97 ;
0000     98 ;       V02-011 REFORMAT        Frederick E. Deen, Jr.      25-Jul-1980
0000     99 ;               This code was reformatted to adhere to RMS standards
0000    100 ;
0000    101 ;       V010    CDS0070 C D Saether     19-Dec-1979
0000    102 ;               Force header write thru even if ACP optimazations are
0000    103 ;               turned on.  This affects RELATIVE, ISAM, and explicit
0000    104 ;               $EXTEND.
0000    105 ;
0000    106 ;       V009    RAN0003 R A Newell      9-Nov-1978
0000    107 ;               File sharing code enhancements
0000    108 ; REVISION HISTORY:
0000    109 ;
0000    110 ;       R A Newell,       9-Nov-1978
0000    111 ;       L F Laverdure,   17-Feb-1978
0000    112 ;       X0001   - File sharing code enhancements
0000    113 ;--
0000    114
```

```
                     0000   116          .SBTTL  DECLARATIONS
                     0000   117
                     0000   118 ;
                     0000   119 ; Include Files:
                     0000   120 ;
                     0000   121
                     0000   122 ;
                     0000   123 ; Macros:
                     0000   124 ;
                     0000   125
                     0000   126          $RMSDEF
                     0000   127          $FIBDEF
                     0000   128          $FABDEF
                     0000   129          $IFBDEF
                     0000   130          $XABDEF
                     0000   131          $XABALLDEF
                     0000   132          $RJRDEF
                     0000   133          $MJBDEF
                     0000   134          $FWADEF
                     0000   135          $ACEDEF
                     0000   136          $CJFDEF
                     0000   137
                     0000   138 ;
                     0000   139 ; Equated Symbols:
                     0000   140 ;
                     0000   141
00000020             0000   142          FOP=FAB$L_FOP*8                 ; bit offset to file options longword
                     0000   143
                     0000   144 ;
                     0000   145 ; Own Storage:
                     0000   146 ;
                     0000   147
```

```
0000    149          .SBTTL  RM$EXTENDO - COMMON FILE EXTEND ROUTINE
0000    150
0000    151 ;++
0000    152 ; RM$EXTENDO - Common file extend routine
0000    153 ;
0000    154 ;  This routine performs common file extension processing
0000    155 ;  including the following:
0000    156 ;
0000    157 ;       1. Allocates a FIB to build the file extension request.
0000    158 ;       2. Initializes the fields of the FIB based upon the type of extend.
0000    159 ;       3. Utilizes the placement information from the XAB, if provided.
0000    160 ;       4. Builds a descriptor for the FIB and calls RM$FCPEXTEND
0000    161 ;          to perform the extend.
0000    162 ;          Write thru is specified to force header write thru so that
0000    163 ;          EOF data will match EOF in PROLOGUE for RELATIVE and ISAM.
0000    164 ;       6. Deallocates the FIB and returns
0000    165 ;
0000    166 ; CALLING SEQUENCE:
0000    167 ;
0000    168 ;       BSBW    RM$EXTENDO
0000    169 ;
0000    170 ;  Alternate entry at RM$EXTENDO_ALT to perform functions 4 & 5 only
0000    171 ;
0000    172 ;       the FIB must already have been allocated and the extend size
0000    173 ;       field filled in.  Address of FIB must be in R1 (R5 and R6 not inputs).
0000    174 ;
0000    175 ;
0000    176 ; INPUT PARAMETERS:
0000    177 ;
0000    178 ;       R11             IMPURE AREA address
0000    179 ;       R10             IFAB address
0000    180 ;       R9              IRAB/IFAB address (IFAB if entry at RM$EXTENDO)
0000    181 ;       R8              RAB/FAB address (FAB if entry at RM$EXTENDO)
0000    182 ;       R6              ALLOCATION XAB address, if any, else 0
0000    183 ;       R5              EXTEND size in blocks
0000    184 ;
0000    185 ; IMPLICIT INPUTS:
0000    186 ;
0000    187 ;       Contents of the FAB
0000    188 ;
0000    189 ; OUTPUT PARAMETERS:
0000    190 ;
0000    191 ;       R6              END VBN of extent + 1
0000    192 ;       R1              STARTING VBN of extent
0000    193 ;       R0              STATUS
0000    194 ;       R2-R5,AP        destroyed
0000    195 ;
0000    196 ; IMPLICIT OUTPUTS:
0000    197 ;
0000    198 ;       None
0000    199 ;
0000    200 ; COMPLETION CODES:
0000    201 ;
0000    202 ;       Standard RMS.
0000    203 ;
0000    204 ; SIDE EFFECTS:
0000    205 ;
```

```
0000   206 ;       May have switched to running at AST level.
0000   207 ;
0000   208 ;--
0000   209
```

```
                     0000   211 RM$EXTENDO::
52   40 8F   9A      0000   212         MOVZBL  #FIB$C_LENGTH,R2                    ; size of FIB
     FFF9'   30      0004   213         BSBW    RM$GETSPC1                          ; allocate FIB
        78   50      E9  0007   214         BLBC    R0,EXIT                             ; get out on error
18 A1    55      D0  000A   215         MOVL    R5,FIB$L_EXSZ(R1)                   ; set extend size
         0B      12  000E   216         BNEQ    10$                                 ; branch if non-zero
18 A1    4C A9   3C  0010   217         MOVZWL  IFB$W_RTDEQ(R9),FIB$L_EXSZ(R1)      ; use default extend size
         04      12  0015   218         BNEQ    10$                                 ; branch if non-zero
16 A1    08      88  0017   219         BISB2   #FIB$M_ALDEF,FIB$W_EXCTL(R1)        ; else use volume default
                     001B   220 10$:
                     001B   221
                     001B   222 ;
                     001B   223 ;  Handle ALLOCATION XAB placement control, if any
                     001B   224 ;
                     001B   225
         56      D5  001B   226         TSTL    R6                                  ; any allocation XAB?
         0F      13  001D   227         BEQL    15$                                 ; branch if none
       0078      30  001F   228         BSBW    RM$SET_XABALL                       ; handle placement control
      27 50      E8  0022   229         BLBS    R0,EXTND                            ; branch if ok
   54    51      D0  0025   230         MOVL    R1,R4                               ; set up regs to return FIB
52   40 8F   9A      0028   231         MOVZBL  #FIB$C_LENGTH,R2                    ;
         5D      11  002C   232         BRB     DEALL_FIB                           ; go deallocate FIB & get out
                     002E   233
                     002E   234 ;
                     002E   235 ;  Set contiguous best try if specified in FOP
                     002E   236 ;
                     002E   237
05 68    35      E1  002E   238 15$:    BBC     #FAB$V_CBT+FOP,(R8),20$             ; branch if CBT bit off
         01      E3  0032   239         BBCS    #FIB$V_ALCONB,-                     ; ask primitive for best try
   11 16 A1                0034   240                 FIB$W_EXCTL(R1),30$          ;  and branch
0D 68    34      E1  0037   241 20$:    BBC     #FAB$V_CTG+FOP,(R8),30$             ; branch if CTG bit off
16 A1    01      88  003B   242         BISB2   #FIB$M_ALCON,FIB$W_EXCTL(R1)        ; ask for contiguous extend
      70 AA      D5  003F   243         TSTL    IFB$L_ABK(R10)                      ; is this first allocation?
         04      12  0042   244         BNEQ    30$                                 ; branch if not
16 A1    04      88  0044   245         BISB2   #FIB$M_FILCON,FIB$W_EXCTL(R1)       ; yes - also mark file CTG
                     0048   246 30$:                                                ; fall thru to RM$EXTENDO_ALT
```

```
                     0048   248
                     0048   249 ;++
                     0048   250 ; RM$EXTEND0_ALT -  Entry point for automatic EXTEND on $PUT.
                     0048   251 ;
                     0048   252 ;       R8, R9 have RAB and IRAB addresses respectively
                     0048   253 ;       FIB must already have been allocated and extend size set, addr in R1.
                     0048   254 ;--
                     0048   255
                     0048   256 RM$EXTEND0_ALT::
   21 A1    03   90  0048   257         MOVB    #FIB$C_VBN,FIB$B_ALALIGN(R1)    ; specify placement near EOF
16 A1    80 8F   88  004C   258 EXTND:  BISB2   #FIB$M_EXTEND,FIB$W_EXCTL(R1)   ; flag this as an EXTEND
                     0051   259         ASSUME  FIB$L_ACCTL EQ 0
                     0051   260         SSB     #FIB$V_WRITETHRU,(R1)           ; force header to write thru
         51    DD   0055   261         PUSHL   R1                              ; build FIB descriptor - addr
   7E    40 8F   9A  0057   262         MOVZBL  #FIB$C_LENGTH,-(SP)             ;                      - length
      FFA2'    30  005B   263         BSBW    RM$FCPEXTEND                    ; do the EXTEND
         14    BA  005E   264         POPR    #^M<R2,R4>                      ; clean STACK and get FIB len & addr
      20 50    E9  0060   265         BLBC    R0,ERREXT                       ; branch if EXTEND failed
                     0063   266
                     0063   267 ;
                     0063   268 ;  EXTEND complete.
                     0063   269 ;
                     0063   270 ;  Write AI journal entry describing successful allocation.  If journal write
                     0063   271 ;  fails, back out the EXTEND with a truncate, and fail the EXTEND.  (If the
                     0063   272 ;  extend isn't backed out, subsequent recovery is in jeopardy because the
                     0063   273 ;  journaled file will not look like the file being recovered, particularly
                     0063   274 ;  because the default extend behavior in recovery could be different from
                     0063   275 ;  extends done here.)
                     0063   276 ;
                     0063   277 ;  Save NEW HIGH VBN + 1 in R6, START VBN in R1, and deallocate FIB
                     0063   278 ;
                     0063   279
                     0063   280 ;
                     0063   281 ; R2 (=size) and R4 (=addr) are input RM$JNL_EXTEND and RM$RETSPC1
                     0063   282 ;
                     0063   283
09 00A0 CA    03   E1  0063   284         BBC     #IFB$V_AI,IFB$B_JNLFLG(R10),10$ ; skip if not AI jnling
    000000F2'EF    16  0069   285         JSB     RM$JNL_EXTEND                   ; journal the extend
         21 50    E9  006F   286         BLBC    R0,JNLERR                       ; branch on error
                     0072   287
         1C A4    DD  0072   288 10$:     PUSHL   FIB$L_EXVBN(R4)                 ; save starting VBN of extent
56    18 A4    6E   C1  0075   289         ADDL3   (SP),FIB$L_EXSZ(R4),R6          ; and END VBN + 1
      FF83'    30  007A   290         BSBW    RM$RETSPC1                      ; deallocate FIB
         02    BA  007D   291         POPR    #^M<R1>                         ; restore STARTING VBN
                     007F   292         RMSSUC
                  05  0082   293 EXIT:   RSB
                     0083   294
                     0083   295 ;++
                     0083   296 ;
                     0083   297 ;  EXTEND failed.
                     0083   298 ;  Map error, deallocate FIB, and return.
                     0083   299 ;
                     0083   300 ;--
                     0083   301
                     0083   302 ERREXT:
                     0083   303         RMSERR  EXT,R1                          ; default status code
      FF75'    30  0088   304 ERRMAP: BSBW    RM$MAPERR                       ; map the error code
```

```
                 008B   305 DEALL_FIB:
      50   DD    008B   306         PUSHL   R0                              ; save it
  FF70'    30    008D   307         BSBW    RM$RETSPC1                      ; deallocate FIB
      01   BA    0090   308         POPR    #^M<R0>                         ; restore status
           05    0092   309         RSB
                 0093   310
                 0093   311 ;++
                 0093   312 ;
                 0093   313 ; Writing journal entry to describe extend failed.  Just eat the extend.
                 0093   314 ; Recovery will notice on the next successful extend that the blocks
                 0093   315 ; allocated are not the same as originally because this extent was eaten.
                 0093   316 ; It will recover by simply extending again.
                 0093   317 ;
                 0093   318 ;--
                 0093   319
                 0093   320 JNLERR:
                 0093   321         RMSERR  CJF,R1                          ; default error message
      EE   11    0098   322         BRB     ERRMAP                          ; deallocate FIB and return
```

```
                                009A   324
                                009A   325 ;++
                                009A   326 ;  RM$SET_XABALL - Handle ALLOCATION XAB placement control,
                                009A   327 ;                 setting up the FIB according to the XAB inputs.
                                009A   328 ;
                                009A   329 ;  INPUTS:
                                009A   330 ;
                                009A   331 ;       R6      XAB address
                                009A   332 ;       R1      FIB address
                                009A   333 ;
                                009A   334 ;  OUTPUTS:
                                009A   335 ;
                                009A   336 ;       R0      STATUS code
                                009A   337 ;       the placement control section of the FIB is initialized.
                                009A   338 ;
                                009A   339 ;  NOTE: No registers other than R0 are modified.
                                009A   340 ;
                                009A   341 ;--
                                009A   342
                                009A   343 RM$SET_XABALL::
05 08 A6    05    E1            009A   344         BBC     #XAB$V_CBT,XAB$B_AOP(R6),20$    ; branch if CBT off
                  01    E3      009F   345         BBCS    #FIB$V_ALCONB,-                 ; ask primitive for contig.
     12 16 A1                   00A1   346                 FIB$W_EXCTL(R1),30$             ; best try and branch
                                00A4   347
                                00A4   348
0D 08 A6    07    E1            00A4   349 20$:    BBC     #XAB$V_CTG,XAB$B_AOP(R6),30$    ; branch if CTG off
   16 A1    01    88            00A9   350         BISB2   #FIB$M_ALCON,FIB$W_EXCTL(R1)    ; ask for contig. extend
         70 A9    D5            00AD   351         TSTL    IFB$L_HBK(R9)                   ; is this first allocation?
                  04    12      00B0   352         BNEQ    30$                             ; branch if not
   16 A1    04    88            00B2   353         BISB2   #FIB$M_FILCON,FIB$W_EXCTL(R1)   ; yes - also mark file CTG
                                00B6   354 30$:
                                00B6   355         ASSUME  XAB$B_ALN EQ XAB$B_AOP+1
                                00B6   356         ASSUME  FIB$B_ALALIGN EQ FIB$B_ALOPTS+1
      00A0 8F    AB            00B6   357         BICW3   #XAB$M_CBT!XAB$M_CTG,-          ; set all. options &
20 A1   08 A6                   00BA   358                 XAB$B_AOP(R6),FIB$B_ALOPTS(R1)  ;  alignment type
                                00BE   359
                                00BE   360
      FC 8F    93               00BE   361         BITB    #^C<XAB$M_HRD!XAB$M_ONC>,-      ; any unknown bits?
      20 A1                     00C1   362                 FIB$B_ALOPTS(R1)
         1B    12               00C3   363         BNEQ    ERRAOP                          ; branch if yes
26 A1    0A A6    B0            00C5   364         MOVW    XAB$W_VOL(R6),FIB$W_LOC_RVN(R1) ; set relative vol. #
   04    09 A6    91            00CA   365         CMPB    XAB$B_ALN(R6),#XAB$C_RFI        ; related file type alloc.?
            07    1F            00CE   366         BLSSU   40$                             ; branch if less
            15    1A            00D0   367         BGTRU   ERRALN                          ; branch if greater
                                00D2   368         ASSUME  FIB$L_LOC_ADDR EQ FIB$W_LOC_FID+6
22 A1    18 A6    7D            00D2   369         MOVQ    XAB$W_RFI(R6),FIB$W_LOC_FID(R1) ; set related FILE ID
      0C A6    D0               00D7   370 40$:    MOVL    XAB$L_LOC(R6),-                 ; set allocation location
      28 A1                     00DA   371                 FIB$L_LOC_ADDR(R1)
                                00DC   372         RMSSUC
               05               00DF   373         RSB
                                00E0   374
                                00E0   375 ;++
                                00E0   376 ;
                                00E0   377 ;  Tell about unknown AOP or ALN values
                                00E0   378 ;
                                00E0   379 ;--
                                00E0   380
```

```
                          00E0   381 ERRAOP:
                          00E0   382          RMSERR  AOP
                    05    00E5   383          BRB     SETSTV
                          00E7   384
                          00E7   385 ERRALN:
                          00E7   386          RMSERR  ALN
 0C A8    17 A6     9A    00EC   387 SETSTV: MOVZBL  XAB$B_AID(R6),FAB$L_STV(R8)     ; area id as STV value
                          00F1   388          RSB
```

```
                     00F2   390          .SBTTL RM$JNL_EXTEND - Journal extend operations
                     00F2   391
                     00F2   392 ;++
                     00F2   393 ; RM$JNL_EXTEND
                     00F2   394 ;
                     00F2   395 ;       This routine is used to journal extend operations.
                     00F2   396 ;
                     00F2   397 ; Calling Sequence:
                     00F2   398 ;
                     00F2   399 ;       BSBW    RM$JNL_EXTEND
                     00F2   400 ;
                     00F2   401 ; Input Parameters:
                     00F2   402 ;
                     00F2   403 ;       R2      size of FIB to journal
                     00F2   404 ;       R4      address of FIB
                     00F2   405 ;       R6      address of XABALL if its present
                     00F2   406 ;
                     00F2   407 ; Implicit Inputs:
                     00F2   408 ;
                     00F2   409 ;       R10     IFB address
                     00F2   410 ;
                     00F2   411 ; Output Parameters:
                     00F2   412 ;
                     00F2   413 ;       R0      status of operation
                     00F2   414 ;       R5      destroyed
                     00F2   415 ;
                     00F2   416 ; Implicit Outputs:
                     00F2   417 ;
                     00F2   418 ;       extend journaled
                     00F2   419 ;
                     00F2   420 ; Side Effects:
                     00F2   421 ;
                     00F2   422 ;       None.
                     00F2   423 ;
                     00F2   424 ;--
                     00F2   425
                     00F2   426 RM$JNL_EXTEND:
                     00F2   427
                     00F2   428          RMSSUC                                  ; anticipate success
    55   34 AA   D0 00F5   429          MOVL    IFB$L_EXTJNLBUF(R10),R5 ; get extend MJB address
         43   13   00F9   430          BEQL    15$                             ; branch if none
     005C 8F   BB 00FB   431          PUSHR   #^M<R2,R3,R4,R6>        ; save work registers
    53   20 A5   DE 00FF   432          MOVAL   MJB$T_RJR(R5),R3        ; get RJR address
37 0A A5   00   E1 0103   433          BBC     #MJB$V_INIT,MJB$W_FLAGS(R5),20$ ; go init RJR if required
                     0108   434 10$:                                            ;
                     0108   435
                     0108   436 ;
                     0108   437 ; Set flags to write extend entry thru to journal and not to give file lock
                     0108   438 ; up during STALL.
                     0108   439 ;
    0A A5   0A   A8 0108   440          BISW2   #<MJB$M_FORCE!MJB$M_SYNCH_SHARE>,MJB$W_FLAGS(R5) ;
                     010C   441
                     010C   442 ;
                     010C   443 ; Copy XABALL fields into RJR if XABALL is present
                     010C   444 ;
                     010C   445 ; These fields are assumed to be in the same order in the XAB and the RJR
                     010C   446 ; Do some ASSUMEs to insure this fact:
```

```
                     010C   447 ;
                     010C   448          ASSUME  XAB$B_ALN       EQ      XAB$B_AOP+1
                     010C   449          ASSUME  RJR$B_EXT_ALN   EQ      RJR$B_EXT_AOP+1
                     010C   450
                     010C   451          ASSUME  XAB$W_VOL       EQ      XAB$B_ALN+1
                     010C   452          ASSUME  RJR$W_EXT_VOL   EQ      RJR$B_EXT_ALN+1
                     010C   453
                     010C   454          ASSUME  XAB$L_LOC       EQ      XAB$W_VOL+2
                     010C   455          ASSUME  RJR$L_EXT_LOC   EQ      RJR$W_EXT_VOL+2
                     010C   456
                     010C   457          ASSUME  XAB$L_ALQ       EQ      XAB$L_LOC+4
                     010C   458          ASSUME  RJR$L_EXT_ALQ   EQ      RJR$L_EXT_LOC+4
                     010C   459
                     010C   460          ASSUME  XAB$W_DEQ       EQ      XAB$L_ALQ+4
                     010C   461          ASSUME  RJR$W_EXT_DEQ   EQ      RJR$L_EXT_ALQ+4
                     010C   462
                     010C   463          ASSUME  XAB$B_AID       EQ      XAB$W_DEQ+3
                     010C   464          ASSUME  RJR$B_EXT_AID   EQ      RJR$W_EXT_DEQ+3
                     010C   465
                     010C   466          ASSUME  XAB$W_RFI       EQ      XAB$B_AID+1
                     010C   467          ASSUME  RJR$W_EXT_RFI   EQ      RJR$B_EXT_AID+1
                     010C   468
          01  23 AA 91  010C   469          CMPB    IFB$B_ORGCASE(R10),#IFB$C_REL ; relative file?
              13 13  0110   470          BEQL    12$                     ; branch if so - ignore possible XAB
              56 D5  0112   471          TSTL    R6                      ; is there an XABALL?
              0F 13  0114   472          BEQL    12$                     ; if EQL no, no XABALL
              3E BB  0116   473          PUSHR   #^M<R1,R2,R3,R4,R5>     ; save regs destroyed by MOVC3
                     0118   474          SSB     #RJR$V_EXT_USE_XAB,RJR$L_EXT_FLAGS(R3) ; say to use XAB
              16 28  011D   475          MOVC3   #<RJR$T_EXT_ENDALL-RJR$B_EXT_AOP>,- ; size of info we want
       64 A3  08 A6  011F   476                  XAB$B_AOP(R6),RJR$B_EXT_AOP(R3) ; fill in RJR from XAB
              3E BA  0123   477          POPR    #^M<R1,R2,R3,R4,R5>     ; restore regs destroyed by MOVC3
              1C A4  C1 0125   478 12$:   ADDL3   FIB$L_EXVBN(R4),-       ; fill in real extend result + 1
              18 A4     0128   479                FIB$L_EXSZ(R4),-
              6C A3     012A   480                RJR$L_EXT_ALQ(R3)
              6C A3  D7 012C   481          DECL    RJR$L_EXT_ALQ(R3)       ; correct value to get real HBK
        00000000'EF  16 012F   482          JSB     RM$WRITE_MJB            ; write the jnl entry
                     0135   483          CSB     #RJR$V_EXT_USE_XAB,RJR$L_EXT_FLAGS(R3) ; re-initialize
            005C 8F  BA 013A   484          POPR    #^M<R2,R3,R4,R6>        ; restore work registers
                     05 013E   485 15$:     RSB                             ; return status to caller
                     013F   486
                     013F   487 20$:
                     013F   488
                     013F   489 ;
                     013F   490 ; Intialize the EXTEND MJB.
                     013F   491 ;
          10 A5   7A 8F  9B 013F   492          MOVZBW  #RJR$C_EXTLEN,MJB$W_SIZE(R5)    ; size of entry to write
              02 A3  01 90 0144   493          MOVB    #RJR$C_VER1,RJR$B_VERSION(R3)   ; rms journaling version
              03 A3  05 90 0148   494          MOVB    #RJR$C_EXTEND,RJR$B_ENTRY_TYPE(R3) ; type is EXTEND
          04 A3   23 AA 90 014C   495          MOVB    IFB$B_ORGCASE(R10),RJR$B_ORG(R3); file organization
              05 A3  0A 90 0151   496          MOVB    #RJR$_EXTEND,RJR$B_OPER(R3)     ; its an EXTEND - superfluous
              51  38 AA D0 0155   497          MOVL    IFB$L_FWA_PTR(R10),R1           ; get FWA address
                 3E BB  0159   498          PUSHR   #^M<R1,R2,R3,R4,R5>     ; save regs destroyed by MOVC3
 08 A3  0920 C1  1C 28  015B   499          MOVC3   #FWA$S_JNLID,FWA$T_JNLID(R1),RJR$T_JNLID(R3) ; set up JNLID
                 3E BA  0162   500          POPR    #^M<R1,R2,R3,R4,R5>     ; restore regs destroyed by MOVC3
            0C A5  03 90 0164   501          MOVB    #CJF$_AI,MJB$B_JNL(R5)          ; set journal type
                     0168   502          SSB     #MJB$V_INIT,MJB$W_FLAGS(R5)     ; indicate MJB initialized
                FF98 31 016D   503          BRW     10$                     ; join common code
```

```
0170   504
0170   505          .END
```

```
$$.PSECT_EP                     =  00000000
$$RMSTEST                       =  0000001A
$$RMS_PBUGCHK                   =  00000010
$$RMS_TBUGCHK                   =  00000008
$$RMS_UMODE                     =  00000004
CJF$_AI                         =  00000003
DEALC_FIB                          0000008B R      01
ERRALN                             000000E7 R      01
ERRAOP                             000000E0 R      01
ERREXT                             00000083 R      01
ERRMAP                             00000088 R      01
EXIT                               00000082 R      01
EXTND                              0000004C R      01
FAB$L_FOP                       =  00000004
FAB$L_STV                       =  0000000C
FAB$V_CBT                       =  00000015
FAB$V_CTG                       =  00000014
FIB$B_ALALIGN                   =  00000021
FIB$B_ALOPTS                    =  00000020
FIB$C_LENGTH                    =  00000040
FIB$C_VBN                       =  00000003
FIB$L_ACCTL                     =  00000000
FIB$L_EXSZ                      =  00000018
FIB$L_EXVBN                     =  0000001C
FIB$L_LOC_ADDR                  =  00000028
FIB$M_ALCON                     =  00000001
FIB$M_ALDEF                     =  00000008
FIB$M_EXTEND                    =  00000080
FIB$M_FILCON                    =  00000004
FIB$V_ALCONB                    =  00000001
FIB$V_WRITETHRU                 =  00000013
FIB$W_EXCTL                     =  00000016
FIB$W_LOC_FID                   =  00000022
FIB$W_LOC_RVN                   =  00000026
FOP                             =  00000020
FWA$S_JNLID                     =  0000001C
FWA$T_JNLID                     =  00000920
IFB$B_JNLFLG                    =  000000A0
IFB$B_ORGCASE                   =  00000023
IFB$C_REL                       =  00000001
IFB$L_EXTJNLBUF                 =  00000034
IFB$L_FWA_PTR                   =  00000038
IFB$L_HBK                       =  00000070
IFB$V_AI                        =  00000003
IFB$W_RTDEQ                     =  0000004C
JNLERR                             00000093 R      01
MJB$B_JNL                       =  0000000C
MJB$M_FORCE                     =  00000002
MJB$M_SYNCH_SHARE               =  00000008
MJB$T_RJR                       =  00000020
MJB$V_INIT                      =  00000000
MJB$W_FLAGS                     =  0000000A
MJB$W_SIZE                      =  00000010
RJR$B_ENTRY_TYPE                =  00000003
RJR$B_EXT_AID                   =  00000073
RJR$B_EXT_ALN                   =  00000065
RJR$B_EXT_AOP                   =  00000064
RJR$B_OPER                      =  00000005
RJR$B_ORG                       =  00000004
RJR$B_VERSION                   =  00000002
RJR$C_EXTEND                    =  00000005
RJR$C_EXTLEN                    =  0000007A
RJR$C_VER1                      =  00000001
RJR$L_EXT_ALQ                   =  0000006C
RJR$L_EXT_FLAGS                 =  0000005C
RJR$L_EXT_LOC                   =  00000068
RJR$T_EXT_ENDALL                =  0000007A
RJR$T_JNLID                     =  00000008
RJR$V_EXT_USE_XAB               =  00000000
RJR$W_EXT_DEQ                   =  00000070
RJR$W_EXT_RFI                   =  00000074
RJR$W_EXT_VOL                   =  00000066
RJR$_EXTEND                     =  0000000A
RMSEXTEND0                         00000000 RG     01
RMSEXTEND0_ALT                     00000048 RG     01
RMSFCPEXTEND                       ********   X    01
RMSGETSPC1                         ********   X    01
RMSJNL_EXTEND                      000000F2 R      01
RMSMAPERR                          ********   X    01
RMSRETSPC1                         ********   X    01
RMSSET_XABALL                      0000009A RG     01
RMSWRITE_MJB                       ********   X    01
RMS$_ALN                        =  000183FC
RMS$_AOP                        =  00018414
RMS$_CJF                        =  0001C164
RMS$_EXT                        =  0001C022
SETSTV                             000000EC R      01
XAB$B_AID                       =  00000017
XAB$B_ALN                       =  00000009
XAB$B_AOP                       =  00000008
XAB$C_RFI                       =  00000004
XAB$L_ALQ                       =  00000010
XAB$L_LOC                       =  0000000C
XAB$M_CBT                       =  00000020
XAB$M_CTG                       =  00000080
XAB$M_HRD                       =  00000001
XAB$M_ONC                       =  00000002
XAB$V_CBT                       =  00000005
XAB$V_CTG                       =  00000007
XAB$W_DEQ                       =  00000014
XAB$W_RFI                       =  00000018
XAB$W_VOL                       =  0000000A
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
RM$RMS0	00000170 (368.)	01 (1.)	PIC USR CON REL GBL NOSHR EXE RD NOWRT NOVEC BYTE
ABS	00000000 (0.)	02 (2.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	29	00:00:00.09	00:00:01.58
Command processing	115	00:00:00.64	00:00:05.26
Pass 1	394	00:00:14.12	00:00:42.01
Symbol table sort	0	00:00:02.14	00:00:04.38
Pass 2	103	00:00:02.65	00:00:07.99
Symbol table output	13	00:00:00.12	00:00:00.12
Psect synopsis output	3	00:00:00.03	00:00:00.03
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	659	00:00:19.79	00:01:01.37

The working set limit was 1500 pages.
79313 bytes (155 pages) of virtual memory were used to buffer the intermediate code.
There were 80 pages of symbol table space allocated to hold 1539 non-local and 16 local symbols.
505 source lines were read in Pass 1, producing 14 object records in Pass 2.
26 pages of virtual memory were used to define 25 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[RMS.OBJ]RMS.MLB;1	14
_255DUA28:[SYS.OBJ]LIB.MLB;1	0
_255DUA28:[SYSLIB]STARLET.MLB;2	7
TOTALS (all libraries)	21

1668 GETS were required to define 21 macros.

There were no errors, warnings or information messages.

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

RM0EXTRMS
LIS

RM0DIRSCN
LIS

RM0CRECOM
LIS

RM0FABCHK
LIS

RM0CHKSUM
LIS

RM0FWASET
LIS

RM0EXTEND
LIS

RM0IFISI
LIS

RM0JOURNL
LIS

RM0FSETI
LIS

RM0FSET
LIS

RM0COMCLN
LIS

RM0DUMMY
LIS

RM0FILFNC
LIS