```
RRRRRRRRRRR      MMM          MMM      SSSSSSSSSSS
RRRRRRRRRRRR     MMM          MMM      SSSSSSSSSSSS
RRRRRRRRRRRR     MMM          MMM      SSSSSSSSSSSS
RRR          RRR MMMMMMM    MMMMMMM   SSS
RRR          RRR MMMMMMM    MMMMMMM   SSS
RRR          RRR MMMMMMM    MMMMMMM   SSS
RRR          RRR MMM   MMM   MMM   MMM SSS
RRR          RRR MMM   MMM   MMM   MMM SSS
RRR          RRR MMM   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               SSS
RRR          RRR MMM          MMM  SSSSSSSSSSS
RRR          RRR MMM          MMM  SSSSSSSSSSS
RRR          RRR MMM          MMM  SSSSSSSSSSS
```

```
RRRRRRRR    MM       MM    000000          JJ    000000    UU       UU  RRRRRRRR    NN       NN  LL
RRRRRRRR    MM       MM    000000          JJ    000000    UU       UU  RRRRRRRR    NN       NN  LL
RR      RR  MMMM   MMMM  00      00        JJ  00      00  UU       UU  RR      RR  NN       NN  LL
RR      RR  MMMM   MMMM  00      00        JJ  00      00  UU       UU  RR      RR  NN       NN  LL
RR      RR  MM MM MM MM  00    0000        JJ  00      00  UU       UU  RR      RR  NNNN     NN  LL
RR      RR  MM MM MM MM  00    0000        JJ  00      00  UU       UU  RR      RR  NNNN     NN  LL
RRRRRRRR    MM  MMM  MM  00  00  00        JJ  00      00  UU       UU  RRRRRRRR    NN NN    NN  LL
RRRRRRRR    MM  MMM  MM  00  00  00        JJ  00      00  UU       UU  RRRRRRRR    NN NN    NN  LL
RR  RR      MM       MM  0000    00  JJ    JJ  00      00  UU       UU  RR  RR      NN   NNNN  LL
RR  RR      MM       MM  0000    00  JJ    JJ  00      00  UU       UU  RR  RR      NN   NNNN  LL
RR    RR    MM       MM  00      00  JJ    JJ  00      00  UU       UU  RR    RR    NN     NN  LL                  ....
RR    RR    MM       MM  00      00  JJ    JJ  00      00  UU       UU  RR    RR    NN     NN  LL                  ....
RR      RR  MM       MM    000000      JJJJJJ    000000    UUUUUUUUUU  RR      RR  NN     NN  LLLLLLLLLL          ....
RR      RR  MM       MM    000000      JJJJJJ    000000    UUUUUUUUUU  RR      RR  NN     NN  LLLLLLLLLL          ....

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

```
0000     1          $BEGIN  RMOJOURNL,000,RM$RMS_JOURNAL,<RMS Journaling Manager>
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:     RMS-32
0000    29 ;
0000    30 ; Abstract:
0000    31 ;               This module provides an interface between RMS and the
0000    32 ;               Common Journaling Facility.
0000    33 ;
0000    34 ; Environment:
0000    35 ;               VAX/VMS Operating System
0000    36 ;
0000    37 ; Author:       Jeffrey W. Horn,                Creation Date:  17-Mar-1982
0000    38 ;
0000    39 ; Modified By:
0000    40 ;
0000    41 ;       V03-044 JWT0162         Jim Teague                 8-Mar-1984
0000    42 ;               Disable RM$RTVJNL for now.
0000    43 ;
0000    44 ;       V03-043 JWT0160         Jim Teague                29-Feb-1984
0000    45 ;               Remove calls to RM$DEALLEFN.
0000    46 ;
0000    47 ;       V03-042 DAS0014         David Solomon             08-Feb-1984
0000    48 ;               Specify ACE$M_NOPROPAGATE for RMSJNLID ACE (they should never
0000    49 ;               be propagated, as they are meaningful to only one file). Fix bug
0000    50 ;               that journal name ACEs were not being marked hidden/protected.
0000    51 ;
0000    52 ;       V03-041 DAS0013         David Solomon             21-Dec-1983
0000    53 ;               Support BRO access for journaling.
0000    54 ;
0000    55 ;       V03-040 JWT0141         Jim Teague                11-Nov-1983
0000    56 ;               Change IFB$V_RUM to IFB$V_ONLY_RU
0000    57 ;
```

```
0000     58 ;       V03-039 KPL0015         Peter Lieberwirth       27-Oct-1983
0000     59 ;               Fix bug introduced in V03-038.  Symptom was breaking relative
0000     60 ;               file extend journaling.
0000     61 ;
0000     62 ;       V03-038 KPL0014         Peter Lieberwirth       20-Oct-1983
0000     63 ;               If doing AI or BI recovery, avoid allocating IRAB JNLBDB
0000     64 ;               and buffer in CONJNL.  This is due to interactions with
0000     65 ;               setting IFB BIO and a recovery process being the only type
0000     66 ;               of process permitted to journal a file open for mixed
0000     67 ;               block and record access (BRO).  Symptom is an FTL$_DEALLER
0000     68 ;               bugcheck because a JNLBDB gets allocated and dropped when
0000     69 ;               another is allocated in RM$WRITE.  (Bugcheck happens on
0000     70 ;               close.)
0000     71 ;
0000     72 ;       V03-037 KPL0013         Peter Lieberwirth       11-Oct-1983
0000     73 ;               Deallocate EFNs after finishing with them.  Improper use
0000     74 ;               of EFNs is causing hangs in asynch situations.  Fix problem
0000     75 ;               with non-page aligned ALDJNLBUF allocations.
0000     76 ;
0000     77 ;       V03-036 DAS0012         David Solomon           27-Sep-1983
0000     78 ;               Preserve R3 in RM$WRTJNL (ISAM assumed it was preserved).
0000     79 ;               Corrected some comments.
0000     80 ;
0000     81 ;       V03-035 DAS0011         David Solomon           08-Sep-1983
0000     82 ;               Correct overzealous fix to RM$DSCJNL in V03-034. Fix test in
0000     83 ;               RM$MAPJNL that decides whether or not this is an open entry.
0000     84 ;               Return RMS$_JNF if no journal name specified, vs RMS$_NOJ.
0000     85 ;
0000     86 ;       V03-034 DAS0010         David Solomon           25-Aug-1983
0000     87 ;               Fix accvio when no journal name is specified. Set up R10 before
0000     88 ;               call to RM$RETJNLBDB (also caused an accvio). Use correct ACE
0000     89 ;               field name for RMS journal names. Replace source.
0000     90 ;
0000     91 ;       V03-033 LJA0090         Laurie J. Anderson      18-Aug-1983
0000     92 ;               1) Fix the writing of the journal entries to not stuff in
0000     93 ;                  the version number as VER1 but rather as the constant
0000     94 ;                  MAXVER so that when the versions are increased (as I
0000     95 ;                  just did) the new version number is filled in.
0000     96 ;               2) Fill in a new (RJR version V04-000 field - for AT journals
0000     97 ;                  the FAB/RAB user CTX field, so that it is written to
0000     98 ;                  the journal for the users discretion.
0000     99 ;               3) Now that the FAB is available when filling in the RJR
0000    100 ;                  use the completion status from it, rather than just
0000    101 ;                  stuff success.
0000    102 ;
0000    103 ;       V03-032 KPL0012         Peter Lieberwirth       30-Jul-1983
0000    104 ;               Allocate a bigger JNLBDB Buffer id AI journaling a relative
0000    105 ;               file.  The larger buffer will be used for the prolog if
0000    106 ;               the file is created.
0000    107 ;
0000    108 ;       V03-031 KPL0011         Peter Lieberwirth       24-Jul-1983
0000    109 ;               Fill in file-oriented AT journal record during MAPJNL
0000    110 ;               call.  Data from IFAB is used to fill in some create/open/close
0000    111 ;               AT fields.  RM$AT_JOURNAL_RECORD fills in some RJR RAB data.
0000    112 ;               RM$AT_COM_RAB added to fill AT record in with initial user
0000    113 ;               search and operation input.
0000    114 ;
```

```
0000   115 ;               Also, fix error paths and block-IO success status path in
0000   116 ;               RM$CONJNL.
0000   117 ;
0000   118 ;               Also, use RM$ALDJNLBUF and RM$RETJNLBDB to allocate and
0000   119 ;               deallocate journaling-specific BDB/Buffers.  Can't just use
0000   120 ;               ALDBUF etc... because then the BDB will be linked into the
0000   121 ;               IFABs BDB list - and could get used for file IO.  Also,
0000   122 ;               now the file-related AT BDB/Buffer can remain allocated for
0000   123 ;               the duration of the file open - previously it was deallocated
0000   124 ;               at common create/open exit because all BDBs on the IFAB
0000   125 ;               list were deallocated at that time.
0000   126 ;
0000   127 ;               Add some commentary about RMS Journaling
0000   128 ;
0000   129 ;       V03-030 KPL0010         Peter Lieberwirth       1-Jul-1983
0000   130 ;               Fix FORCE_JNL to always return status.
0000   131 ;
0000   132 ;       V03-029 KPL0009         Peter Lieberwirth       16-Jun-1983
0000   133 ;               Fix some bugs.  Add routine to write AT journal records for
0000   134 ;               record operations.  Clean up RM$MAPJNL to let it write AT
0000   135 ;               file operation records.  Remove COP and CQE in favor of CJF.
0000   136 ;               Move misc IFAB jnl flags to JNLFLG2.
0000   137 ;
0000   138 ;       V03-028 TSK0052         Tamar Krichevsky        5-jun-1983
0000   139 ;               Fix bugs introduced by V03-26.  Move module to RM$RMS_JOURNAL
0000   140 ;               psect.  Fix broken branches to RM$MAPERR.
0000   141 ;
0000   142 ;       V03-027 KPL0008         Peter Lieberwirth       30-May-1983
0000   143 ;               Fix bugs introduced in V03-026 and earlier.
0000   144 ;
0000   145 ;       V03-026 KPL0007         Peter Lieberwirth       26-May-1983
0000   146 ;               Support new more robust RJR format.  Fix typos in KPL0001.
0000   147 ;               Turn on sequential file journaling.  Rework RJB/BDB allocation.
0000   148 ;
0000   149 ;       V03-025 TSK0050         Tamar Krichevsky        25-May-1983
0000   150 ;               Modify RM$CONJNL to allocate the proper size journal buffer
0000   151 ;               for sequential files.  Currently, the user specified bucket
0000   152 ;               size is used to determine the buffer's length.  For sequential
0000   153 ;               files, the buffer must be large enough to contain any one
0000   154 ;               record from the file.
0000   155 ;               Cleanup calculation of overhead for journal buffer.
0000   156 ;
0000   157 ;       V03-024 DAS0009         David Solomon           11-May-1983
0000   158 ;               Fix WRTACC check in RM$ASSJNL (BBC to BBS). Add missing "#"
0000   159 ;               in front of two literals that were causing accvio's.  Fix
0000   160 ;               error path on failure to assign channel to RU journal.  Clear
0000   161 ;               pointer to RJB upon its deallocation.  Don't allocate IRAB
0000   162 ;               AT journal buffer if not AT journaling.  Fix ALLOC_MJB to
0000   163 ;               acquire space from same page as IFAB.  Do better job at
0000   164 ;               calculating required size of MJB.
0000   165 ;
0000   166 ;       V03-023 KPL0006         Peter Lieberwirth       2-May-1983
0000   167 ;               Turn on $WRITEJNL call.  Add $WRMODDEF.  Fix bug on
0000   168 ;               error path into RM$DEAJNL.
0000   169 ;
0000   170 ;       V03-022 KPL0005         Peter Lieberwirth       1-May-1983
0000   171 ;               Delete obsolete MJB definitions.
```

```
0000   172 ;
0000   173 ;       V03-021 KPL0004         Peter Lieberwirth       1-May-1983
0000   174 ;               Fix another problem with $WRITEJNL call.
0000   175 ;
0000   176 ;       V03-020 KPL0003         Peter Lieberwirth       1-May-1983
0000   177 ;               Fix call to $WRITEJNL.
0000   178 ;
0000   179 ;       V03-019 KPL0002         Peter Lieberwirth      30-Apr-1983
0000   180 ;               Add omitted macro definition.  Flesh out WRITE_MJB
0000   181 ;               routine.
0000   182 ;
0000   183 ;       V03-018 KPL0001         Peter Lieberwirth      29-Apr-1983
0000   184 ;               Allocate miscellaneous journaling buffers for IFB and IRB
0000   185 ;               where necessary.  Generalize cleanup so these always get
0000   186 ;               deallocated.  Add stub RM$WRITE_MJB routine.
0000   187 ;
0000   188 ;       V03-017 JWH0221         Jeffrey W. Horn         26-Apr-1983
0000   189 ;               If in recovery allow BRO access.  Also temporarily, enable
0000   190 ;               both AI and BI journaling durring recovery.
0000   191 ;
0000   192 ;       V03-016 JWH0205         Jeffrey W. Horn         11-Apr-1983
0000   193 ;               Implement journal id ACE.  Also add protected and hidden
0000   194 ;               bits to all ACEs.
0000   195 ;
0000   196 ;       V03-015 DAS0008         David Solomon           01-Apr-1983
0000   197 ;               Save R2 in RM$WRTJNL (for ISAM).
0000   198 ;
0000   199 ;       V03-014 RAS0135         Ron Schaefer            17-Mar-1983
0000   200 ;               More corrections to RAS0132 for registers and RJR$_ names.
0000   201 ;
0000   202 ;       V03-013 RAS0135         Ron Schaefer            17-Mar-1983
0000   203 ;               Corrections to RAS0132 for registers and RJR$_ names.
0000   204 ;
0000   205 ;       V03-012 RAS0132         Ron Schaefer            16-Mar-1983
0000   206 ;               Merge $RMSRDEF into $RJRDEF and revise the interface
0000   207 ;               for RM$WRTJNL for easier use from ISAM.
0000   208 ;
0000   209 ;       V03-011 JWH0185         Jeffrey W. Horn         11-Feb-1983
0000   210 ;               Set WRFLG$V_BI on RU journal entries.
0000   211 ;               Use the perm FWA to provide journal entry security and
0000   212 ;               to fill in the mapping entries.
0000   213 ;               If file is opened UFO then disable journaling for this open.
0000   214 ;
0000   215 ;       V03-010 JWH0180         Jeffrey W. Horn         03-Feb-1983
0000   216 ;               Change references to RJR$C_MAPLEN from byte to word.
0000   217 ;
0000   218 ;       V03-009 JWH0173         Jeffrey W. Horn         24-Jan-1983
0000   219 ;               Clean up status code returns.
0000   220 ;               Use BKS instead of MRS to allocate journal BDB.
0000   221 ;               Allow ISAM journaling.
0000   222 ;
0000   223 ;       V03-008 JWH0167         Jeffrey W. Horn         10-Jan-1983
0000   224 ;               Implement IFB recovery option byte.
0000   225 ;               Fill in file organization in mapping entry.
0000   226 ;
0000   227 ;       V03-007 JWH0155         Jeffrey W. Horn          3-Dec-1982
0000   228 ;               Seperate journal names into three seperate ACEs.
```

```
0000   229 ;               Prevent journaling on Sequential and Indexed files.
0000   230 ;               For block io, do not create journal BDB and buffer.
0000   231 ;
0000   232 ;       V03-006 JWH0154         Jeffrey W. Horn         13-Dec-1982
0000   233 ;               Define ACE$C_JNLNAMS (temporary).
0000   234 ;
0000   235 ;       V03-005 JWH0132         Jeffrey W. Horn         22-Nov-1982
0000   236 ;               Write journal entries with the WRFLG$M_LOCK attribute.
0000   237 ;
0000   238 ;       V03-004 JWH0128         Jeffrey W. Horn         15-Nov-1982
0000   239 ;               Change SS$_NOCJF code to SS$_IVSSRQ.
0000   240 ;
0000   241 ;       V03-003 JWH0116         Jeffrey W. Horn         28-Oct-1982
0000   242 ;               If in RCP then don't perfom any journaling execpt AT.
0000   243 ;               Remove CALLS to CJF services and replace with macros.
0000   244 ;               Change logic in FRCJNL which checks for an active RU to
0000   245 ;               reflect changes in RUF.
0000   246 ;
0000   247 ;       V03-002 JWH0108         Jeffrey W. Horn         23-Sep-1982
0000   248 ;               Remove redefinitions of ACL ACP attributes.
0000   249 ;               Fix problem with setting size for RJB deallocation.
0000   250 ;               Clean up status code returns.
0000   251 ;               Redefine journal names (FWA$T_xxJNLN) as .ASCIC
0000   252 ;               strings.
0000   253 ;               Implement new RMS journaling record (RJR).
0000   254 ;               Use RM$GETBLK and RM$RETBLK instead of RM$GETSPC and
0000   255 ;               RM$RETSPC when allocating and deallocting the RJB.
0000   256 ;
0000   257 ;       V03-001 JWH0107         Jeffrey W. Horn         23-Sep-1982
0000   258 ;               Redefine ACL ACP attributes to ATR$C_USERLABEL which is a
0000   259 ;               no-op.  Add a .WEAK for CJF$GETJNL.  Clean up status code
0000   260 ;               returns.
0000   261 ;
0000   262 ;--
```

```
0000    264          .SBTTL  DECLARATIONS
0000    265
0000    266
0000    267 ;
0000    268 ; Include Files:
0000    269 ;
0000    270
0000    271 ;
0000    272 ; Macros:
0000    273 ;
0000    274
0000    275          $ACEDEF
0000    276          $ATRDEF
0000    277          $BDBDEF
0000    278          $CJFDEF
0000    279          $DVIDEF
0000    280          $FABDEF
0000    281          $RABDEF
0000    282          $FIBDEF
0000    283          $FWADEF
0000    284          $IFBDEF
0000    285          $IODEF
0000    286          $IMPDEF
0000    287          $IRBDEF
0000    288          $PCBDEF
0000    289          $PSLDEF
0000    290          $RJBDEF
0000    291          $RJRDEF
0000    292          $RMSDEF
0000    293          $RUCBDEF
0000    294          $SSDEF
0000    295          $STSDEF
0000    296          $WRFLGDEF
0000    297          $MJBDEF
0000    298          $WRMODDEF
0000    299
0000    300 ;
0000    301 ; Equated Symbols:
0000    302 ;
0000    303
0000    304 ;
0000    305 ; Own Storage:
0000    306 ;
0000    307
0001 0000    308 FACILITY:       .WORD   RMS$_FACILITY
0001 0002    309 MODE:           .WORD   PSL$C_EXEC
```

```
0004   311          .SUBTITLE Introduction to RMS Journaling
0004   312 ;++
0004   313 ;       RMS Journaling Manager
0004   314 ;
0004   315 ; This module contains routines used to journal RMS operations.  Other modules
0004   316 ; containing journaling routines (not necessarily an inclusive list) are:
0004   317 ;
0004   318 ;       RM3JOURNL.B32, RM1JOURNL.MAR, RMOCRECOM.MAR, RMOBUFMGR.MAR,
0004   319 ;       RMOEXTEND.MAR, and RM2CREATE.MAR
0004   320 ;
0004   321 ; The data structures are defined in:
0004   322 ;
0004   323 ;       RMSINTSTR.MDL and the format of the RMS Journaling Record (RJR) is
0004   324 ; described in RMSFILSTR.SDL.
0004   325 ;
0004   326 ; The general flow of journaling control is as follows:
0004   327 ;
0004   328 ; 1. When a file marked for journaling is accessed, connections are made
0004   329 ;    to the journals specified in the file's header in RM$ASSJNL.  Certain
0004   330 ;    data structures are allocated at this time also.
0004   331 ;
0004   332 ; 1a. If the file is being created, the data structures are allocated earlier,
0004   333 ;    and the JNLXAB is interrogated for journal names.  If no journal names
0004   334 ;    are specified in the XAB, CJF is asked for default journal names.  This
0004   335 ;    is done in RM$GETJNL.
0004   336 ;
0004   337 ; 2. RM$MAPJNL is called to write entries to the journals at OPEN/CREATE/CLOSE
0004   338 ;    time.  These entries contain the full filename and other information.
0004   339 ;    These entries are used when the journal must be interrogated for file
0004   340 ;    names, and to associate a filename with a journal ID.
0004   341 ;
0004   342 ;       A journal ID is a unique identifier associated with a journaled file
0004   343 ;       (it is kept in the file header in a hidden, protected, access control
0004   344 ;       entry).  It is used in most RMS journaling records so that the full
0004   345 ;       filename need not be kept in all entries.  It is also used as a
0004   346 ;       short-hand identifier to search a journal for RMS entries without
0004   347 ;       having to fully specify the filename as originally journaled.
0004   348 ;
0004   349 ;
0004   350 ; 3. RM$CONJNL is called at connect time to allocate record-oriented RMS
0004   351 ;    journaling structures.  These include buffers and buffer descriptors.
0004   352 ;    These structures are deallocated at disconnect time in RM$DSCJNL.
0004   353 ;    RM$DSCJNL also forces to the journal any audit-trail journal entries
0004   354 ;    written to CJF but not yet necessarily forced to the actual journal
0004   355 ;    (IE the entries may still be in a CJF buffer.)
0004   356 ;
0004   357 ; 4. During the course of RMS record operations journal entries describing
0004   358 ;    file accesses and modifications are written to the appropriate journals.
0004   359 ;
0004   360 ;       ISAM AI and BI operations are journaled by writing copies of the
0004   361 ;       modified buckets to the journal.  The buffers used for these entries
0004   362 ;       are as follows:
0004   363 ;
0004   364 ;               AI - the buffer used is the actual data bucket that is written
0004   365 ;                    to the file
0004   366 ;
0004   367 ;               BI - the buffer used is an extra one allocated at the same time
```

```
0004   368 ;                    the data buffer is allocated
0004   369 ;
0004   370 ;                Both buffers are pointed to by the BDB.
0004   371 ;
0004   372 ;        ISAM AI and BI operations are journaled at the bucket-level because
0004   373 ;        there was no way found to journal on a record basis and ensure that
0004   374 ;        RFAs would be restored upon recovery.
0004   375 ;
0004   376 ;        ISAM recovery unit operations are journaled by writing information
0004   377 ;        describing the modified record to the journal.  The ISAM code treats
0004   378 ;        record operations in recovery units in a special fashion:
0004   379 ;
0004   380 ;                $DELETEs do not delete the record - the record is merely
0004   381 ;                marked for deletion.
0004   382 ;
0004   383 ;                $UPDATEs never shrink the size of the record - extra space
0004   384 ;                corresponding to the original size of the record is kept
0004   385 ;                and described by special fields in the record itself.
0004   386 ;
0004   387 ;                The reason for never deleting space in ISAM RUs is to ensure
0004   388 ;                there will always be space in the bucket if the record
0004   389 ;                must be rolled back in.  We don't want to invent more
0004   390 ;                special case ISAM bucket split code.  The RFA basis of the
0004   391 ;                journal entry also precludes too much bucket entropy before
0004   392 ;                recovery.
0004   393 ;
0004   394 ;        Sequential and Relative file journaling is done on a record basis.
0004   395 ;        A record journaling buffer is allocated at CONNECT time, and this
0004   396 ;        buffer is used to build the record used to describe the change needed
0004   397 ;        to undo or redo the operation.
0004   398 ;
0004   399 ;        Audit-trail journaling is done on a file and record level.  A special
0004   400 ;        BDB and Buffer is allocated off the IFAB to contain file related
0004   401 ;        audit-trail information.  A journaling buffer descriptor/buffer
0004   402 ;        is allocated off the IRAB to collect and format record-related
0004   403 ;        audit trail information.
0004   404 ;
0004   405 ;        In order to ensure ISAM AI recovery, $EXTENDs must be journaled.
0004   406 ;        A special extend buffer descriptor/buffer is allocated off the
0004   407 ;        IFAB - the journaling record to describe the extend is built in
0004   408 ;        and written from this buffer.  Sequential and Relative AI extends
0004   409 ;        are journaled in the same fashion.
0004   410 ;
0004   411 ; 5. RMS Journaling Data Structures
0004   412 ;
0004   413 ;        RJB     - The RJB is allocated by ASSJNL or CRECOM, and contains
0004   414 ;                  the channels assigned to various journals.  Flags indicating
0004   415 ;                  connections to journals are also present.
0004   416 ;
0004   417 ;        IFB JNLFLG - This byte is a copy of the file header byte which
0004   418 ;                  indiates what types of journaling the file is marked for.
0004   419 ;
0004   420 ;        IFB JNLFLG2 - This byte contains miscellaneous run-time IFAB related
0004   421 ;                  journaling indicators.
0004   422 ;
0004   423 ;        IFB$L_JNLBDB - This field points to a BDB and buffer that is used for
0004   424 ;                  file related AI journaling.
```

```
0004   425 ;
0004   426 ;       IFB$L_ATJNLBUF - This field points into the buffer pointed to indirectly
0004   427 ;              by IFB$L_JNLBDB.  This field points directly to the RJR within
0004   428 ;              the buffer.
0004   429 ;
0004   430 ;       RJR - RMS Journaling Record.  The format of the RMS data written to
0004   431 ;              the journal.  It is comprised of a common overhead, and several
0004   432 ;              different formats following the common overhead that are used
0004   433 ;              for different journaling functions.
0004   434 ;
0004   435 ;              Currently implemented:  FILE, RECORD, BLOCK, BUCKET, EXTEND,
0004   436 ;              AT_RECORD.
0004   437 ;
0004   438 ;       MJB - Miscellaneous Journaling Block  This is used to describe
0004   439 ;              miscellaneous journaling records and the information needed
0004   440 ;              to describe the WRITEJNL request.  The MJB is written by
0004   441 ;              RM$WRITE_MJB and is forced to the journal by RM$FORCE_MJB.
0004   442 ;
0004   443 ;              MJBs are currently used for AT and Extend entries.
0004   444 ;
0004   445 ;       IRB$L_ATJNLBUF - points to an MJB/Buffer used to write record level AT
0004   446 ;              entries.
0004   447 ;
0004   448 ;       Why MJBs and BDBs?  Good question.  The BDB related design is good for
0004   449 ;       writing buffers containing actual file data to the journals.  The
0004   450 ;       MJB is used when descriptive entries not directly related to file
0004   451 ;       data are written.  BDB/Buffer fits into the IO system concept and
0004   452 ;       ISAM AI and BI benefits from the overlap.  MJB/Buffer fits into
0004   453 ;       the CJF design better.  The MJB describes the WRITEJNL inputs,
0004   454 ;       basically.  The only counter-intuitive setup currently is writing
0004   455 ;       file-level descriptive entries via BDB and not MJB.  The reason for this
0004   456 ;       is that MAPJNL was originally set up this way.
0004   457 ;
0004   458 ;--
```

```
                        0004    460          .SBTTL  RM$GETJNL - Get Journal Name
                        0004    461
                        0004    462 ;++
                        0004    463 ; RM$GETJNL - Get Journal Name
                        0004    464 ;
                        0004    465 ;       This subroutines gets the journal names to use from either CJF
                        0004    466 ;       or the process-based default journal names.  It then proceeds to
                        0004    467 ;       set up the attributes for the file creation.
                        0004    468 ;
                        0004    469 ;
                        0004    470 ; Calling sequence:
                        0004    471 ;
                        0004    472 ;       BSBW    RM$GETJNL
                        0004    473 ;
                        0004    474 ; Input Parameters:
                        0004    475 ;
                        0004    476 ;       R9      -       IFAB address
                        0004    477 ;       R10     -       FWA address
                        0004    478 ;
                        0004    479 ; Implicit Inputs:
                        0004    480 ;
                        0004    481 ;       IFB$B_JNLFLG -  File's Journaling Flags
                        0004    482 ;       FWA$L_UIC    -  File's Owner UIC
                        0004    483 ;       FWA$Q_xxJNL, FWA$W_xxJNLN - may be preset by XAB processing to contain
                        0004    484 ;                                   some journal names.
                        0004    485 ;
                        0004    486 ; Output Parameters:
                        0004    487 ;
                        0004    488 ;       R1-R4           Destroyed
                        0004    489 ;
                        0004    490 ; Implicit Outputs:
                        0004    491 ;
                        0004    492 ;       FWA$Q_xxJNL, FWA$Q_xxJNLN - Set to journal name(s).
                        0004    493 ;
                        0004    494 ; Completion Codes:
                        0004    495 ;
                        0004    496 ;       JNF - If no journal name found for a particular IFB$B_JNLFLG bit,
                        0004    497 ;                 STV will contain CJF status from $GETJNL.
                        0004    498 ;
                        0004    499 ; Side Effects:
                        0004    500 ;       None.
                        0004    501 ;
                        0004    502 ;--
                        0004    503
                        0004    504 RM$GETJNL::
      7E    01    D0    0004    505          MOVL    #1,-(SP)                        ; anticipate success
16 00A0 C9    02    E1    0007    506          BBC     #IFB$V_BI,IFB$B_JNLFLG(R9),10$  ; branch if no BI bit
   52   08C8 CA    9E    000D    507          MOVAB   FWA$Q_BIJNL(R10),R2             ; fwa bi descr
   53   08E0 CA    9E    0012    508          MOVAB   FWA$T_BIACE(R10),R3             ; fwa bi buffer
        54   02    D0    0017    509          MOVL    #CJF$_BI,R4                     ; journal type code
           0084    30    001A    510          BSBW    GET_JNL                         ; get journal name
          03 50    E8    001D    511          BLBS    R0,10$                          ; get out on error
        6E   50    D0    0020    512          MOVL    R0,(SP)                         ; remember error code
                        0023    513
16 00A0 C9    03    E1    0023    514 10$:     BBC     #IFB$V_AI,IFB$B_JNLFLG(R9),20$  ; branch if no AI bit
   52   08D0 CA    9E    0029    515          MOVAB   FWA$Q_AIJNL(R10),R2             ; fwa AI descr
   53   08F4 CA    9E    002E    516          MOVAB   FWA$T_AIACE(R10),R3             ; fwa AI buffer
```

```
                     54    03    D0  0033    517          MOV     #CJF$_AI,R4                             ; journal type code
                        0068    30  0036    518          BSBW    GET_JNL                                 ; get journal name
                       03 50    E8  0039    519          BLBS    R0,20$                                  ; get out on error
                     6E    50    D0  003C    520          MOVL    R0,(SP)                                 ; remember error code
                                     003F    521
           16 00A0   C9    04    E1  003F    522 20$:     BBC     #IFB$V_AT,IFB$B_JNLFLG(R9),30$          ; branch if no AT bit
              52     08D8  CA    9E  0045    523          MOVAB   FWA$Q_ATJNL(R10),R2                     ; fwa AT descr
              53     0908  CA    9E  004A    524          MOVAB   FWA$T_ATACE(R10),R3                     ; fwa AT buffer
                     54    04    D0  004F    525          MOVL    #CJF$_AT,R4                             ; journal type code
                        004C    30  0052    526          BSBW    GET_JNL                                 ; get journal name
                       03 50    E8  0055    527          BLBS    R0,30$                                  ; continue on success
                     6E    50    D0  0058    528          MOVL    R0,(SP)                                 ; remember error code
                                     005B    529
           092C CA   01F8  CA    D0  005B    530 30$:     MOVL    <FWA$T_FIBBUF+FIB$W_FID>(R10),FWA$T_FID(R10) ; put fid in id ace
           0930 CA   01FC  CA    B0  0062    531          MOVW    <FWA$T_FIBBUF+FIB$W_FID+4>(R10),<FWA$T_FID+4>(R10)
                                     0069    532          $GETTIM_S TIMADR=FWA$Q_ID_DATE(R10)             ; get current time
        091C CA    0E000820 8F    D0  0074    533          MOVL    #<<<ACE$M_PROTECTED + ACE$M_HIDDEN + ACE$M_NOPROPAGATE> -
                                     007D    534                  @ <ACE$W_FLAGS*8>> + -
                                     007D    535                    <ACE$C_JNLID @ <ACE$B_TYPE*8>> + -
                                     007D    536                    FWA$S_IDACE>, FWA$T_IDACE(R10)
                     85    20    B0  007D    537          MOVW    #FWA$S_IDACE,(R5)+                      ; set attribute len
                     85    1F    B0  0080    538          MOVW    #ATR$C_ADDACLENT,(R5)+                  ; set attribute type
                  85    091C  CA    DE  0083    539          MOVAL   FWA$T_IDACE(R10),(R5)+                  ; set attribute address
                                     0088    540          RMSSUC
                                     008B    541
                     50    8E    D0  008B    542 50$:     MOVL    (SP)+,R0                                ; get status code
                       01 50    E9  008E    543          BLBC    R0,60$                                  ; skip if error
                                     05  0091    544          RSB
                                     0092    545
                  00A0 C9    94    0092    546 60$:     CLRB    IFB$B_JNLFLG(R9)                        ; turn off journaling
                                     0096    547          RMSERR  JNF,RT                                  ; journal not found
           00000000'EF    17    009B    548          JMP     RMS$MAPERR                              ; go map the error and retur
```

```
                     00A1    550          .SBTTL  GET_JNL - Common Get Journal name routine
                     00A1    551
                     00A1    552 ;++
                     00A1    553 ; GET_JNL - Common Get Journal name routine
                     00A1    554 ;
                     00A1    555 ; If XAB processing did not get a particular journal name, then ask
                     00A1    556 ; CJF for one.
                     00A1    557 ;
                     00A1    558 ; Calling sequence:
                     00A1    559 ;
                     00A1    560 ;       BSBW    GET_JNL
                     00A1    561 ;
                     00A1    562 ; Input Parameters:
                     00A1    563 ;
                     00A1    564 ;       R2      -       Pointer to FWA$Q_xxJNL  (fwa journal name descriptor)
                     00A1    565 ;       R3      -       Pointer to FWA$T_xxJNLN (fwa journal name buffer)
                     00A1    566 ;       R4      -       CJF$_xx for the journal type
                     00A1    567 ;       R5      -       Address of first free slot at end of ACP attribute list
                     00A1    568 ;
                     00A1    569 ; Implicit Inputs:
                     00A1    570 ;
                     00A1    571 ;       FWA$L_UIC       File Ownership UIC.
                     00A1    572 ;       FWA$Q_DEVICE    Descriptor of Device name
                     00A1    573 ;       FWA$L_ATR_LIST  Atribute list for create
                     00A1    574 ;
                     00A1    575 ; Output Parameters:
                     00A1    576 ;       R5              New free ACP attribute list free slot.
                     00A1    577 ;
                     00A1    578 ; Implicit Outputs:
                     00A1    579 ;
                     00A1    580 ;       FWA$Q_xxJNL, FWA$T_xxJNLN - filled in
                     00A1    581 ;       FWA$T_ATR_LIST - May have journal name attributes added.
                     00A1    582 ;
                     00A1    583 ; Completion Codes:
                     00A1    584 ;       Any CJF from $GETJNL.
                     00A1    585 ;
                     00A1    586 ; Side Effects:
                     00A1    587 ;       None.
                     00A1    588 ;--
                     00A1    589
                     00A1    590 GET_JNL:
                     00A1    591
                     00A1    592 ;
                     00A1    593 ; If no journal name from XAB processing, ask CJF for one
                     00A1    594 ;
          7E    01   D0  00A1    595         MOVL    #1,-(SP)                                ; assume success
                62   95  00A4    596         TSTB    (R2)                                    ; name length zero?
                32   12  00A6    597         BNEQ    20$                                     ; no branch
          62    10   3C  00A8    598         MOVZWL  #FWA$S_BIJNLN,(R2)                      ; set up descriptor
    04 A2     04 A3   DE  00AB    599         MOVAL   ACE$T_RMSJNLNAM(R3),4(R2)
             28 AA    D5  00B0    600         TSTL    FWA$L_UIC(R10)                          ; file uic specified?
                0D   12  00B3    601         BNEQ    10$                                     ; branch if so
  51    00000000'9F   D0  00B5    602         MOVL    a#CTL$GL_PCB,R1                         ; get PCB address
   28 AA    00BC C1   D0  00BC    603         MOVL    PCB$L_UIC(R1),FWA$L_UIC(R10)            ; get UIC from PCB
                          00C2    604
                          00C2    605 10$:    $GETJNL_S       -                               ; call CJF
                          00C2    606                 DEVNAM = FWA$Q_DEVICE(R10), -
```

```
                                00C2    607                     UIC     = FWA$L_UIC(R10), -
                                00C2    608                     JNLTYP  = R4, -
                                00C2    609                     JNLNAM  = (R2), -
                                00C2    610                     RSLLEN  = (R2)
                                00D7    611
                    6E    50    D0    00D7    612             MOVL    R0,(SP)                                 ; save return code
                                00DA    613
                                00DA    614 ;
                                00DA    615 ; Construct ACE to store journal name and add to attribute list
                                00DA    616 ;
                                00DA    617             ASSUME  ACE$C_BIJNL EQ CJF$_BI
                                00DA    618             ASSUME  ACE$C_AIJNL EQ <ACE$C_BIJNL + 1>
                                00DA    619             ASSUME  ACE$C_ATJNL EQ <ACE$C_AIJNL + 1>
                                00DA    620
                63  62    04    81    00DA    621 20$:        ADDB3   #ACE$T_RMSJNLNAM,(R2),(R3)              ; fill in ACE size
                  01  A3    54    90    00DE    622             MOVB    R4,ACE$B_TYPE(R3)                       ; move type into ACE
                      0600  8F    B0    00E2    623             MOVW    #ACE$M_HIDDEN!ACE$M_PROTECTED,-         ; move flags into ACE
                        02  A3          00E6    624                     ACE$W_FLAGS(R3)
                      85    63    9B    00E8    625             MOVZBW  (R3),(R5)+                              ; move atr len into list
                      85    1F    B0    00EB    626             MOVW    #ATR$C_ADDACLENT,(R5)+                  ; move atr type into list
                      85    53    D0    00EE    627             MOVL    R3,(R5)+                                ; move atr addr into list
                      50    8E    D0    00F1    628             MOVL    (SP)+,R0                                ; restore code
                                05    00F4    629             RSB
```

```
                     00F5   631          .SBTTL  RM$RTVJNL - Retrieve Journaling Info
                     00F5   632 ;++
                     00F5   633 ; RM$RTVJNL - Retrieve Journaling Info
                     00F5   634 ;
                     00F5   635 ;       This subroutine adds the neccessary ACP attributes to retrieve
                     00F5   636 ;       both the journal selection bits and the journal names used for a file.
                     00F5   637 ;
                     00F5   638 ; Calling Sequence:
                     00F5   639 ;
                     00F5   640 ;       BSBW    RM$RTVJNL
                     00F5   641 ;
                     00F5   642 ; Input Parameters
                     00F5   643 ;       R5      Address of End of attribute list
                     00F5   644 ;       R9      IFAB address
                     00F5   645 ;       R10     FWA Address
                     00F5   646 ;       R11     Impure Area Address
                     00F5   647 ;
                     00F5   648 ; Implicit Imputs:
                     00F5   649 ;       None.
                     00F5   650 ;
                     00F5   651 ; Ouput Parameters:
                     00F5   652 ;
                     00F5   653 ;       R1      Destroyed
                     00F5   654 ;       R5      Updated to new end of attribute list
                     00F5   655 ;
                     00F5   656 ; Implicit Outputs:
                     00F5   657 ;
                     00F5   658 ;       FWA ACP attribute list has attributes filled in to retrieve journaling
                     00F5   659 ;       bits and journal names.
                     00F5   660 ;
                     00F5   661 ; Completion Codes:
                     00F5   662 ;       None.
                     00F5   663 ;
                     00F5   664 ; Side Effects:
                     00F5   665 ;       None.
                     00F5   666 ;
                     00F5   667 ;--
                     00F5   668
                     00F5   669 RM$RTVJNL::
                     00F5   670
                     00F5   671 ;**JNL** begin temporary code to tie off journaling
                05   00F5   672          RSB
                     00F6   673 ;**JNL** end temporary code to tie off journaling
                     00F6   674
                     00F6   675 ;
                     00F6   676 ; Construct ACEs to get journal names and add ACP attribute
                     00F6   677 ;
51   08E0 CA   DE    00F6   678          MOVAL   FWA$T_BIACE(R10),R1                             ; get start of ACE
61   0214 8F   B0    00FB   679          MOVW    #<<ACE$C_BIJNL@<ACE$B_TYPE*8>>+FWA$S_BIACE>,(R1) ; move in ACE Type,
     85   14   B0    0100   680          MOVW    #FWA$S_BIACE,(R5)+                              ; move atr len into list
     85   23   B0    0103   681          MOVW    #ATR$C_FNDACLTYP,(R5)+                          ; move atr type into list
     85   51   D0    0106   682          MOVL    R1,(R5)+                                        ; move atr addr into list
                     0109   683
51   08F4 CA   DE    0109   684          MOVAL   FWA$T_AIACE(R10),R1                             ; get start of ACE
61   0314 8F   B0    010E   685          MOVW    #<<ACE$C_AIJNL@<ACE$B_TYPE*8>>+FWA$S_AIACE>,(R1) ; move in ACE Type,
     85   14   B0    0113   686          MOVW    #FWA$S_AIACE,(R5)+                              ; move atr len into list
     85   23   B0    0116   687          MOVW    #ATR$C_FNDACLTYP,(R5)+                          ; move atr type into list
```

```
                85   51   D0  0119   688          MOVL    R1,(R5)+                                ; move atr addr into list
                                 011C   689
          51    0908 CA   DE  011C   690          MOVAL   FWA$T_ATACE(R10),R1                     ; get start of ACE
          61    0414 8F   B0  0121   691          MOVW    #<<ACE$C_ATJNL@<ACE$B_TYPE*8>>+FWA$S_ATACE>,(R1) ; move in ACE Type,
                85   14   B0  0126   692          MOVW    #FWA$S_ATACE,(R5)+                      ; move atr len into list
                85   23   B0  0129   693          MOVW    #ATR$C_FNDACLTYP,(R5)+                  ; move atr type into list
                85   51   D0  012C   694          MOVL    R1,(R5)+                                ; move atr addr into list
                                 012F   695
          51    091C CA   DE  012F   696          MOVAL   FWA$T_IDACE(R10),R1                     ; get start of ACE
    61     00000820 8F   D0  0134   697          MOVL    #<<ACE$C_JNLID@<ACE$B_TYPE*8>>+FWA$S_IDACE>,(R1) ; set up ACE
                85   20   B0  013B   698          MOVW    #FWA$S_IDACE,(R5)+                      ; move atr len into list
                85   23   B0  013E   699          MOVW    #ATR$C_FNDACLTYP,(R5)+                  ; move atr type into list
                85   51   D0  0141   700          MOVL    R1,(R5)+                                ; move atr addr into list
                                 0144   701
                                 0144   702 ;
                                 0144   703 ; Add journal control bit attributes to list
                                 0144   704 ;
                85   01   B0  0144   705          MOVW    #1,(R5)+                                ; move atr len into list
                85   1D   B0  0147   706          MOVW    #ATR$C_JOURNAL,(R5)+                    ; move atr type into list
          85    00A0 C9   9E  014A   707          MOVAB   IFB$B_JNLFLG(R9),(R5)+                  ; move atr addr into list
                                 014F   708
                                 014F   709 ;
                                 014F   710 ; Make sure we have the file's UIC in the FWA
                                 014F   711 ;
                85   04   B0  014F   712          MOVW    #4,(R5)+                                ; move atr len into list
                85   1A   B0  0152   713          MOVW    #ATR$C_UIC_RO,(R5)+                     ; move atr type into list
            85    28 AA   DE  0155   714          MOVAL   FWA$L_UIC(R10),(R5)+                    ; move atr addr into list
                                 0159   715
                         05  0159   716          RSB
```

```
                     015A   718          .SBTTL  RM$ASSJNL - Open Journaling for a file
                     015A   719
                     015A   720 ;++
                     015A   721 ; RM$ASSJNL - Open Journaling for a file
                     015A   722 ;
                     015A   723 ;       This subroutine builds the necessary data structures for journaling
                     015A   724 ;       onto the IFAB and opens the journals needed for the file.
                     015A   725 ;
                     015A   726 ; Calling sequence:
                     015A   727 ;
                     015A   728 ;       BSBW    RM$ASSJNL
                     015A   729 ;
                     015A   730 ; Input Parameters:
                     015A   731 ;
                     015A   732 ;       R8      FAB Address
                     015A   733 ;       R9      IFAB Address
                     015A   734 ;       R10     FWA Address
                     015A   735 ;       R11     Impure Area Address
                     015A   736 ;
                     015A   737 ; Implicit Inputs:
                     015A   738 ;
                     015A   739 ;       IFB$B_JNLFLG
                     015A   740 ;
                     015A   741 ; Output Parameters:
                     015A   742 ;
                     015A   743 ;       R1 - R5 Destroyed
                     015A   744 ;
                     015A   745 ; Implicit Outputs:
                     015A   746 ;
                     015A   747 ;       IFB$L_RJB       Address of allocated and initialized RJB
                     015A   748 ;       IFB$B_JNLFLG2   Files Journaling Flags:
                     015A   749 ;               IFB$V_JNL       Set to indicate journaling initialized for this
                     015A   750 ;                               file.
                     015A   751 ;
                     015A   752 ; Completion Codes:
                     015A   753 ;
                     015A   754 ;       Any RMS, particularly, DME.
                     015A   755 ;       NOJ, Journal device for file not available, CJF status in
                     015A   756 ;               STV from $ASSJNL.
                     015A   757 ;       JNS, Journaling not supported for operation
                     015A   758 ;
                     015A   759 ; Side Effects:
                     015A   760 ;       None.
                     015A   761 ;
                     015A   762 ;--
                     015A   763
                     015A   764 ERRJNS: RMSERR  JNS
                  05 015F   765          RSB
                     0160   766
          00A0 C9    94 0160   767 UFO:     CLRB    IFB$B_JNLFLG(R9)                        ; turn off journaling
                     0164   768 ASS_DONE:
                     0164   769          RMSSUC
                  05 0167   770          RSB
                     0168   771
                     0168   772 RM$ASSJNL::
F6 00A2 C9    04    E2 0168   773          BBSS    #IFB$V_DONE_ASS_JNL,IFB$B_JNLFLG2(R9),ASS_DONE ; already thru
                     016E   774                                                          ; here during $CREATE.
```

```
                ED 04 A8    11    E0  016E   775          BBS     #FAB$V_UFO,FAB$L_FOP(R8),UFO                ; branch if UFO
                07 22 A9    05    E1  0173   776          BBC     #IFB$V_BIO,IFB$B_FAC(R9),10$                ; branch if not BIO
                 00A0 C9    03    93  0178   777          BITB    #<IFB$M_RU!IFB$M_ONLY_RU>,IFB$B_JNLFLG(R9)      ; don't allow RU BIO
                            DB    12  017D   778          BNEQ    ERRJNS
                                      017F   779
                                      017F   780 ;
                                      017F   781 ; Next, if the process in which we're executing is a RECOVERY process we
                                      017F   782 ; may not want to journal.  Specifically, if the file we're starting to
                                      017F   783 ; access is one RMS Recovery is recovering, we don't want to
                                      017F   784 ;
                                      017F   785 ;       a. recovery unit journal
                                      017F   786 ;       b. AI or BI journal if we're doing AI recovery
                                      017F   787 ;
                                      017F   788 ; Note: BI recovery must be journaled.  If BI recovery is not journaled,
                                      017F   789 ; the file can be in states never represented by any state representable
                                      017F   790 ; by the RMS journal entries in the journal.  This can happen when a file
                                      017F   791 ; is BI journaled, modified, rolled-back, modified again, and later rolled
                                      017F   792 ; back to a time when first modified.  This is because 'old' record images
                                      017F   793 ; are put in BI journals.  Therefore, a record may get put in the file that
                                      017F   794 ; never shows up in the journal.  Therefore if its backed out by Recovery,
                                      017F   795 ; and recovery is not journaled - that record will never be seen again.
                                      017F   796 ; This problem does not occur with AI journaling because the journal contains
                                      017F   797 ; 'new' record images.
                                      017F   798 ;
                                      017F   799
          51   00000000'9F    D0    017F   800 10$:     MOVL    @#CTL$GL_PCB,R1                             ; get PCB address for test
                16 24 A1     1A    E1  0186   801          BBC     #PCB$V_RECOVER,PCB$L_STS(R1),20$            ; skip rest if not
                                      018B   802                                                              ;  in RECOVER
                       00A1 C9     95  018B   803          TSTB    IFB$B_RECVRFLGS(R9)                         ; may be in RECOVER, but
                                      018F   804                                                              ;  not recovering this
                                      018F   805                                                              ;  file
                            10    13  018F   806          BEQL    20$                                         ; branch if not in recovery
                                      0191   807
                00A0 C9     03    8A  0191   808          BICB    #<IFB$M_RU!IFB$M_ONLY_RU>,IFB$B_JNLFLG(R9)      ; clear RU journalin
          05 00A1 C9        01    E1  0196   809          BBC     #IFB$V_AI_RECVR,IFB$B_RECVRFLGS(R9),20$     ; skip next if not AI
                00A0 C9     0C    8A  019C   810          BICB    #<IFB$M_AI!IFB$M_BI>,IFB$B_JNLFLG(R9)       ; clear AI, BI if AI
                                      01A1   811
                  07 69     30    E0  01A1   812 20$:     BBS     #IFB$V_WRTACC,(R9),50$                      ; branch if writing
                00A0 C9     0F    8A  01A5   813          BICB    #<IFB$M_AI!IFB$M_BI!IFB$M_RU!IFB$M_ONLY_RU>,IFB$B_JNLFLG(R9)
                                      01AA   814                                                              ; clear AI,BI,RU
                            50    11  01AA   815          BRB     3000$                                       ; branch to AI test.
                                      01AC   816
                                      01AC   817 50$:                                                        ;
          06 00A0 C9        00    E1  01AC   818 60$:     BBC     #IFB$V_ONLY_RU,IFB$B_JNLFLG(R9),1000$       ; branch if ONLY_RU
                                      01B2   819          SSB     #IFB$V_RU,IFB$B_JNLFLG(R9)                  ; set RU bit
                                      01B8   820
          13 00A0 C9        02    E1  01B8   821 1000$:  BBC     #IFB$V_BI,IFB$B_JNLFLG(R9),2000$            ; branch if no BI
                53    08C8 CA     7E  01BE   822          MOVAQ   FWA$Q_BIJNL(R10),R3                         ; BI descriptor
                54    08E0 CA     9E  01C3   823          MOVAB   FWA$T_BIACE(R10),R4                         ; BI name
                      55    02    D0  01C8   824          MOVL    #CJF$_BI,R5                                 ; indicate BI
                         009B     30  01CB   825          BSBW    OPEN_JNL                                    ; go open channel
                       67 50      E9  01CE   826          BLBC    R0,5000$                                    ; get out on error
                                      01D1   827
          25 00A0 C9        03    E1  01D1   828 2000$:  BBC     #IFB$V_AI,IFB$B_JNLFLG(R9),3000$            ; branch if no AI
                52    009A 8F     3C  01D7   829          MOVZWL  #<MJB$C_BLN+RJR$C_EXTLEN>,R2                ; size of MJB for extend
                 000006AA'EF      16  01DC   830          JSB     RM$ALLOC_MJB                                ; get the MJB
                       53 50      E9  01E2   831          BLBC    R0,5000$                                    ; get out on error
```

```
                34 A9    51    D0  01E5    832           MOVL    R1,IFB$L_EXTJNLBUF(R9)                  ; set up pointer
             53    08D0 CA    7E  01E9    833           MOVAQ   FWA$Q_AIJNL(R10),R3                     ; AI descriptor
             54    08F4 CA    9E  01EE    834           MOVAB   FWA$T_AIACE(R10),R4                     ; AI name
                   55    03    D0  01F3    835           MOVL    #CJF$_AI,R5                             ; indicate AI
                      0070    30  01F6    836           BSBW    OPEN_JNL                                ; go open channel
                     3C 50    E9  01F9    837           BLBC    R0,5000$                                ; get out on error
                              01FC    838
          13 00A0 C9    04    E1  01FC    839 3000$:    BBC     #IFB$V_AT,IFB$B_JNLFLG(R9),4000$        ; branch if no AT
             53    08D8 CA    7E  0202    840           MOVAQ   FWA$Q_ATJNL(R10),R3                     ; AT descriptor
             54    0908 CA    9E  0207    841           MOVAB   FWA$T_ATACE(R10),R4                     ; AT name
                   55    04    D0  020C    842           MOVL    #CJF$_AT,R5                             ; indicate AT
                      0057    30  020F    843           BSBW    OPEN_JNL                                ; go open channel
                     23 50    E9  0212    844           BLBC    R0,5000$                                ; get out on error
                              0215    845
          4A 00A0 C9    01    E1  0215    846 4000$:    BBC     #IFB$V_RU,IFB$B_JNLFLG(R9),6000$        ; branch if no RU
                   55    01    D0  021B    847           MOVL    #CJF$_RU,R5                             ; indicate RU
                      0048    30  021E    848           BSBW    OPEN_JNL                                ; go open channel
                     14 50    E9  0221    849           BLBC    R0,5000$                                ; return on success
       51    00000000'9F    D0  0224    850           MOVL    a#CTL$GL_RUF,R1                         ; already in RU?
                        38    13  022B    851           BEQL    6000$                                   ; branch if not
             36 11 A1    01    E1  022D    852           BBC     #RUCB$V_ACTIVE,RUCB$B_CTRL(R1),7000$
          30 00A2 C9    02    E3  0232    853           BBCS    #IFB$V_RUF,IFB$B_JNLFLG2(R9),7000$      ; set RU in prog
                              0238    854                                                           ; NOTE: Should never
                              0238    855                                                           ; fall through
                              0238    856
                   00A0 C9    94  0238    857 5000$:    CLRB    IFB$B_JNLFLG(R9)                        ; on error clr flgs
       51    50    0C    10    EF  023C    858           EXTZV   #STS$V_FAC_NO,#STS$S_FAC_NO,R0,R1       ; get error facility
                   51    01    D1  0241    859           CMPL    #RMS$_FACILITY,R1                       ; is error from RMS?
                        22    13  0244    860           BEQL    7000$                                   ; don't map if so
                   52    50    D0  0246    861           MOVL    R0,R2                                   ; save CJF status
             00000000'EF    16  0249    862           JSB     RM$MAPERR                               ; fill in STV
       52    00000000'8F    D1  024F    863           CMPL    #CJF$_NONAME,R2                         ; was error no jnl name?
                        07    12  0256    864           BNEQ    5010$                                   ; no, use NOJ error
                              0258    865           RMSERR  JNF                                     ; yes, use JNF error
                        05    11  025D    866           BRB     5020$                                   ;  and continue
                              025F    867 5010$:    RMSERR  NOJ                                     ; use NOJ error
                        05    0264    868 5020$:    RSB                                             ; return
                              0265    869
                              0265    870 6000$:    RMSSUC                                          ; yes, indicate success
                              0268    871
                        05    0268    872 7000$:    RSB
```

```
0269   874          .SBTTL  OPEN_JNL - Common open journal channel
0269   875
0269   876 ;++
0269   877 ; OPEN_JNL - Common open journal channel
0269   878 ;
0269   879 ; This routine opens a channel on the specified journal.  It also alocates
0269   880 ; an RJB if needed.
0269   881 ;
0269   882 ; Calling sequence:
0269   883 ;
0269   884 ;       BSBW    OPEN_JNL
0269   885 ;
0269   886 ; Input Parameters:
0269   887 ;
0269   888 ;       R3      Address of Journal Name Descriptor (FWA$Q_xxJNL) (AI,BI,AT only)
0269   889 ;       R4      Address of Journal Name ACE (FWA$T_xxACE) (AI,BI,AT only)
0269   890 ;       R5      Journal Type (CJF$_xx)
0269   891 ;       R9      IFAB address
0269   892 ;       R10     FWA address
0269   893 ;       R11     Impure area address
0269   894 ;
0269   895 ; Implicit Inputs:
0269   896 ;
0269   897 ;       IFB$L_RJB       RJB address
0269   898 ;       IFB$B_JNLFLG    File's journaling flags
0269   899 ;       FWA$Q_DEVICE    Device file resides on.
0269   900 ;       FWA$Q_xxJNL, FWA$T_xxJNLN
0269   901 ;                       Journal Names for file
0269   902 ;       FWA$L_UIC       File Owner
0269   903 ;       FWA$L_PRO       File Protection
0269   904 ;
0269   905 ; Output Parameters:
0269   906 ;
0269   907 ;       R1-R5           Destroyed
0269   908 ;
0269   909 ; Implicit Outputs:
0269   910 ;
0269   911 ;       IFB$L_RJB       Address of allocated RJB
0269   912 ;       IFB$B_JNLFLG2   Files Journaling flags
0269   913 ;           IFB$V_JNL       Set to indicate journaling initialized.
0269   914 ;       RJB$W_FLAGS     A bit is set for each channel opened.
0269   915 ;       RJB$Q_CHAN      One word is filled in with a channel number.
0269   916 ;
0269   917 ; Completion Codes:
0269   918 ;
0269   919 ;       Any RMS, particualrly, DME,
0269   920 ;       Any CJF status value from $ASSJNL.
0269   921 ;
0269   922 ;
0269   923 ; Side Effects:
0269   924 ;
0269   925 ;       If journaling not previosly initialized on this file, allocates an RJB
0269   926 ;       for it.
0269   927 ;
0269   928 ;--
0269   929
0269   930 OPEN_JNL:
```

```
                0559   30  0269   931          BSBW    RM$ALLOC_RJB_BDB                        ; get journaling BDB/Buffer
              03 50    E8  026C   932          BLBS    R0,10$                                  ; continue if success
                007D   31  026F   933          BRW     50$                                     ; out on error
          52  00A4 C9   D0  0272   934 10$:     MOVL    IFB$L_RJB(R9),R2                        ; get RJB address
              01   55   D1  0277   935          CMPL    R5,#CJF$_RU                             ; Opening RU?
                   3A   13  027A   936          BEQL    20$                                     ; yes, branch
                   63   D4  027C   937          CLRL    (R3)                                    ; set up descriptor
          63   64   04   83  027E   938          SUBB3   #ACE$T_RMSJNLNAM,(R4),(R3)              ; get length of journal name
                   09   14  0282   939          BGTR    15$                                     ; length is >0
       50  00000000'8F   D0  0284   940          MOVL    #CJF$_NONAME,R0                         ; journal not specified
                   58   11  028B   941          BRB     40$                                     ; error exit
       04 A3    04 A4     DE  028D   942 15$:     MOVAL   ACE$T_RMSJNLNAM(R4),4(R3)               ; fill in address of string
                          0292   943
                          0292   944          $ASSJNL_S       -                                   ; assign journal chan
                          0292   945                  CHAN = RJB$Q_CHAN-2(R2)[R5], -
                          0292   946                  JNLTYP = R5, -
                          0292   947                  JNLNAM = (R3), -
                          0292   948                  ACMODE = MODE, -
                          0292   949                  PROT   = FWA$W_PRO(R10), -
                          0292   950                  OBJUIC = FWA$L_UIC(R10), -
                          0292   951                  FACCOD = FACILITY
                          02B4   952
                   24     11  02B4   953          BRB     30$
                          02B6   954
                          02B6   955 20$:     $ASSJNL_S       -                                   ; open RU chan
                          02B6   956                  CHAN = RJB$Q_CHAN(R2), -
                          02B6   957                  JNLTYP = R5, -
                          02B6   958                  DEVNAM = FWA$C_DEVICE(R10), -
                          02B6   959                  ACMODE = MODE, -
                          02B6   960                  PROT   = FWA$W_PRO(R10), -
                          02B6   961                  OBJUIC = FWA$L_UIC(R10), -
                          02B6   962                  FACCOD = FACILITY
                          02DA   963
          08 50    E9     02DA   964 30$:     BLBC    R0,40$                                  ; return on error
                          02DD   965
             55    D7     02DD   966          DECL    R5                                      ; one less than type
                          02DF   967          SSB     R5,RJB$W_FLAGS(R2)                      ; turn on bit for chan
                   05     02E4   968          RSB                                             ; return to caller
                          02E5   969
                          02E5   970 ;
                          02E5   971 ; Error Exit
                          02E5   972 ;
                   01     BB  02E5   973 40$:     PUSHR   #^M<R0>                                 ; save R0
  000005F2'        EF     16  02E7   974          JSB     RM$DEAJNL                               ; deallocate RJB
                   01     BA  02ED   975          POPR    #^M<R0>                                 ; restore R0
                          02EF   976
                   05     02EF   977 50$:     RSB
```

```
                     02F0   979          .SBTTL  RM$CONJNL - Connect Journal BDB
                     02F0   980
                     02F0   981 ;++
                     02F0   982 ; RM$CONJNL - Connect Journal BDB
                     02F0   983 ;
                     02F0   984 ; This routine, called from $CONNECT, builds the necessary data
                     02F0   985 ; structures onto the IRAB for journaling record processing
                     02F0   986 ; operations
                     02F0   987 ;
                     02F0   988 ; Calling sequence:
                     02F0   989 ;
                     02F0   990 ;       BSBW    RM$CONJNL
                     02F0   991 ;
                     02F0   992 ; Input Parameters:
                     02F0   993 ;
                     02F0   994 ;       R9      Address of IRAB
                     02F0   995 ;       R10     Address of IFAB
                     02F0   996 ;       R11     Address of Impure area
                     02F0   997 ;
                     02F0   998 ; Implicit Inputs:
                     02F0   999 ;
                     02F0  1000 ;       None.
                     02F0  1001 ;
                     02F0  1002 ; Output Parameters:
                     02F0  1003 ;
                     02F0  1004 ;       R1 - R3,R5      Destroyed
                     02F0  1005 ;       R4              Address of BDB for journaling I/O.
                     02F0  1006 ;
                     02F0  1007 ; Implicit Outputs:
                     02F0  1008 ;
                     02F0  1009 ;       IRB$L_JNLBDB    Address of BDB for journaling I/O.
                     02F0  1010 ;
                     02F0  1011 ; Completion Codes:
                     02F0  1012 ;       Any valid RMS, particualarly DME.
                     02F0  1013 ;
                     02F0  1014 ; Side Effects:
                     02F0  1015 ;       A buffer and BDB are allocated, the BDB is marked perm.
                     02F0  1016 ;
                     02F0  1017 ;--
                     02F0  1018
                     02F0  1019 RM$CONJNL::
                     02F0  1020
                     02F0  1021 ;
                     02F0  1022 ; Determine whether or not we need to allocate a journal BDB and buffer. We
                     02F0  1023 ; only need one if connecting for record access. For block I/O access, simply
                     02F0  1024 ; exit (the journal BDB and buffer will be allocated on the first $WRITE).
                     02F0  1025 ;
                     02F0  1026
        05   E0      02F0  1027          BBS     #IFB$V_BIO,-            ; if we're open for BIO, exit
0A 22 AA             02F2  1028                  IFB$B_FAC(R10),10$      ;
        06   E1      02F5  1029          BBC     #IFB$V_BRO,-            ; if not opening BRO, we're ok
08 22 AA             02F7  1030                  IFB$B_FAC(R10),20$      ;  (must be open for record access)
        0B   E1      02FA  1031          BBC     #RAB$V_BIO,-            ; if connecting for record access,
03 04 A8             02FC  1032                  RAB$L_ROP(R8),20$       ;  we're ok
     007C    31      02FF  1033 10$:     BRW     80$                     ; exit
                     0302  1034
                     0302  1035 ;
```

```
                     0302  1036 ; If the file is sequential, determine the largest probable record size to be
                     0302  1037 ; journaled.  A record can be no larger than the maximum record length.  If
                     0302  1038 ; the MRS was not given, then look at the the longest record length or the
                     0302  1039 ; multiblock count.  If none of these values were specified, then punt.
                     0302  1040 ;
                     0302  1041
                     0302  1042         ASSUME  IFB$C_SEQ EQ 0
                     0302  1043
        23 AA   95   0302  1044 20$:    TSTB    IFB$B_ORGCASE(R10)              ; is the file sequential?
           1D   12   0305  1045         BNEQ    50$                             ; no, use BKS for buffer len
                     0307  1046
55      60 AA   3C   0307  1047         MOVZWL  IFB$W_MRS(R10),R5               ; use the max rec. size
           1F   12   030B  1048         BNEQ    60$                             ; use it if present
                     030D  1049
55      52 AA   3C   030D  1050         MOVZWL  IFB$W_LRL(R10),R5               ; use the LRL for the buffer
           19   12   0311  1051         BNEQ    60$                             ; finish buffer size calulat
                     0313  1052
55      37 A8   9A   0313  1053         MOVZBL  RAB$B_MBC(R8),R5                ; use the MBC for buffer len
           04   13   0317  1054         BEQL    30$                             ; no, buffer will be 1 page
           67   19   0319  1055         BLSS    ERRMBC                          ; MBC must be > 0
           0B   11   031B  1056         BRB     55$
                     031D  1057
55    0200 8F   3C   031D  1058 30$:    MOVZWL  #512,R5                         ; buff. will be 1 page
           08   11   0322  1059         BRB     60$
                     0324  1060
                     0324  1061 ;
                     0324  1062 ; file is not sequential.  Use the bucket size as the buffer length.
                     0324  1063 ;
                     0324  1064
     55   5E AA   9A   0324  1065 50$:    MOVZBL  IFB$B_BKS(R10),R5               ; get bucket size
55   55   09   78   0328  1066 55$:    ASHL    #9,R5,R5                        ; convert to bytes
                     032C  1067
55   00000048 8F   C0   032C  1068 60$:    ADDL2   #RJR$C_RECLEN, R5               ; give some overhead
55   000001FF 8F   C0   0333  1069         ADDL2   #511,R5                         ; round up to a
55   000001FF 8F   CA   033A  1070         BICL2   #511,R5                         ; page boundary
                     0341  1071
     00000000'EF   16   0341  1072         JSB     RM$ALDJNLBUF                    ; get BDB and buffer
           37 50   E9   0347  1073         BLBC    R0,90$                          ; get out on error
              3E   BB   034A  1074         PUSHR   #^M<R1,R2,R3,R4,R5>             ; save regs zeroed by MOVC5
     51   18 A4   D0   034C  1075         MOVL    BDB$L_ADDR(R4),R1               ; get RJR address
61   38   00   61   00   2C   0350  1076         MOVC5   #0,(R1),#0,#RJR$C_HDRLEN,(R1)   ; zero the RJR overhead
              3E   BA   0356  1077         POPR    #^M<R1,R2,R3,R4,R5>             ; restore regs zeroed by MOV
        30 A9   54   D0   0358  1078         MOVL    R4,IRB$L_JNLBDB(R9)             ; save BDB addr
                     035C  1079
                     035C  1080         ASSUME  RJR$C_EXTLEN    GT      RJR$C_BLKLEN
                     035C  1081         ASSUME  RJR$C_EXTLEN    GT      RJR$C_AT_RECLEN
                     035C  1082
 1C 00A0 CA   04   E1   035C  1083         BBC     #IFB$V_AT,IFB$B_JNLFLG(R10),80$ ; skip if not AT
52   009A 8F   3C   0362  1084         MOVZWL  #<MJB$C_BLN+RJR$C_EXTLEN>,R2    ; length of structure
     02   23 AA   91   0367  1085         CMPB    IFB$B_ORGCASE(R10),#IFB$C_IDX   ; indexed file?
              07   12   036B  1086         BNEQ    70$                             ; if NEQ no
52   00000100 8F   C0   036D  1087         ADDL    #256,R2                         ; add in max key size
                0333   30   0374  1088 70$:    BSBW    RM$ALLOC_MJB                    ; allocate MJB
              07 50   E9   0377  1089         BLBC    R0,90$                          ; branch if error
        2C A9   51   D0   037A  1090         MOVL    R1,IRB$L_ATJNLBUF(R9)           ; init pointer
                     037E  1091 80$:    RMSSUC                                  ; indicate success
                     0381  1092 90$:    RSB
```

```
                0382  1093
                0382  1094 ERRMBC:
                0382  1095          RMSERR  MBC
          05    0387  1096          RSB
```

```
                     0388  1098          .SBTTL  RM$MAPJNL - Write Mapping Entry
                     0388  1099
                     0388  1100 ;++
                     0388  1101 ; RM$MAPJNL - Write Mapping Entry
                     0388  1102 ; RM$MAPJNL_RU - Write RU Mapping Entry
                     0388  1103 ;
                     0388  1104 ; This routine writes a mapping entry into all currently open
                     0388  1105 ; journals for a particular file
                     0388  1106 ;
                     0388  1107 ; Calling sequence:
                     0388  1108 ;
                     0388  1109 ;       BSBW    RM$MAPJNL
                     0388  1110 ;       BSBW    RM$MAPJNL_RU
                     0388  1111 ;
                     0388  1112 ; Input Parameters·
                     0388  1113 ;
                     0388  1114 ;       R8      FAB address (used by COMMON_FILE_AT to write CTX field into RJR)
                     0388  1115 ;       R9      IFAB address
                     0388  1116 ;       R11     Impure area address
                     0388  1117 ;       AP      r0 status till now (I know its a hack, but..) only used for AT
                     0388  1118 ;
                     0388  1119 ; Implicit Inputs:
                     0388  1120 ;
                     0388  1121 ;       IFB$L_RJB       RJB address
                     0388  1122 ;       IFB$L_FWA_PTR   FWA pointer and current contents of FWA
                     0388  1123 ;       RJB$V_OPEN      Set to indicate an open entry; cleared if set.
                     0388  1124 ;       RJB$W_FLAGS     RMS journal channel flags - these will be used
                     0388  1125 ;                       as variable inputs (saved and restored by caller)
                     0388  1126 ;                       to allow AT write at a different time from AI, BI, RU.
                     0388  1127 ;
                     0388  1128 ; Output Parameters:
                     0388  1129 ;
                     0388  1130 ;       R1 - R5         Destroyed
                     0388  1131 ;
                     0388  1132 ; Implicit Outputs:
                     0388  1133 ;
                     0388  1134 ;       RJB$V_OPEN      Cleared if set
                     0388  1135 ;
                     0388  1136 ; Completion Codes:
                     0388  1137 ;
                     0388  1138 ;       Any RMS, particularly DME,
                     0388  1139 ;       CJF -   CJF error, CJF status in STV
                     0388  1140 ;
                     0388  1141 ; Side Effects:
                     0388  1142 ;       May have switched to EXEC AST level.
                     0388  1143 ;
                     0388  1144 ;--
                     0388  1145
                     0388  1146 ;
                     0388  1147 ; Alternate Entry Point for RU handler
                     0388  1148 ;
                     0388  1149
                     0388  1150 RM$MAPJNL_RU::
01   DD              0388  1151          PUSHL   #1                                  ; indicate RU MAPJNL
02   11              038A  1152          BRB     MAPJNL
                     038C  1153
                     038C  1154 ;
```

```
                     038C  1155 ; Entry point for AI, BI, AT
                     038C  1156 ;
                     038C  1157 RMSMAPJNL::
           7E   D4   038C  1158         CLRL    -(SP)                           ; indicate not RU MAPJNL
                     038E  1159
                     038E  1160
     7E    56   7D   038E  1161 MAPJNL: MOVQ    R6,-(SP)                        ; save R6, R7
     7E    5A   D0   0391  1162         MOVL    R10,-(SP)                       ; save R10
                     0394  1163
                     0394  1164 ;
                     0394  1165 ; Get RJR buffer address.
                     0394  1166 ;
        042E   30    0394  1167         BSBW    RMSALLOC_RJB_BDB                ; get a journal BDB
                     0397  1168                                                 ; if this is CLOSE
      03 50   E8     0397  1169         BLBS    R0,10$                          ; continue if OK
       009C   31     039A  1170         BRW     80$                             ; out on error
  5A   30 A9   D0    039D  1171 10$:    MOVL    IFB$L_JNLBDB(R9),R10            ; first get BDB address
  56   18 AA   D0    03A1  1172         MOVL    BDB$L_ADDR(R10),R6              ; get RJR address
                     03A5  1173
                     03A5  1174 ;
                     03A5  1175 ; Fill in file name in entry
                     03A5  1176 ;
  5A   38 A9   D0    03A5  1177         MOVL    IFB$L_FWA_PTR(R9),R10           ; get FWA address
53   00C4 C6   DE    03A9  1178         MOVAL   RJR$T_FILENAME(R6),R3           ; get name buff addr
                     03AE  1179
                     03AE  1180         ASSUME  RJR$S_FILENAME EQ 256
                     03AE  1181
                     03AE  1182 ;
                     03AE  1183 ; Set buffer size to 255 because the GETFILNAM code builds a NAM block, etc...
                     03AE  1184 ; and can only cope with a size that fits in a byte.
                     03AE  1185 ;
54   00FF 8F   3C    03AE  1186         MOVZWL  #<RJR$S_FILENAME-1>,R4          ; set size of buffer
 00000000'EF   16    03B3  1187         JSB     RMSGETFILNAM                    ; go get file name
   58 A6   54   90    03B9  1188         MOVB    R4,RJR$B_FNS(R6)                ; put length in entry
                     03BD  1189
                     03BD  1190 ; Fill in header
                     03BD  1191 ;
     54   30 A9   D0    03BD  1192         MOVL    IFB$L_JNLBDB(R9),R4             ; retrieve jnl BDB addr
14 A4   01C4 8F   B0    03C1  1193         MOVW    #RJR$C_FILNAMLEN,BDB$W_NUMB(R4) ; set entry size
   57   00A4 C9   D0    03C7  1194         MOVL    IFB$L_RJB(R9),R7                ; get RJB address
   03 A6   01   90    03CC  1195         MOVB    #RJR$C_MAPPING,RJR$B_ENTRY_TYPE(R6) ; fill in file type
04 A6   23 A9   90    03D0  1196         MOVB    IFB$B_ORGCASE(R9),RJR$B_ORG(R6) ; fill in org
        0C AE   D5    03D5  1197         TSTL    ^X0C(SP)                        ; RU call?
           52   12    03D8  1198         BNEQ    70$                             ; branch if so
                     03DA  1199
                     03DA  1200         ASSUME  FAB$C_SEQ@-4    EQ      RJR$C_SEQ
                     03DA  1201         ASSUME  FAB$C_REL@-4    EQ      RJR$C_REL
                     03DA  1202         ASSUME  FAB$C_IDX@-4    EQ      RJR$C_IDX
                     03DA  1203
06 0A A7   04   E5    03DA  1204         BBCC    #RJB$V_OPEN,RJB$W_FLAGS(R7),20$ ; branch if not $OPEN
   05 A6   11   90    03DF  1205         MOVB    #RJR$_OPEN,RJR$B_OPER(R6)       ; fill in operation
           04   11    03E3  1206         BRB     30$
                     03E5  1207
   05 A6   02   90    03E5  1208 20$:    MOVB    #RJR$_CLOSE,RJR$B_OPER(R6)      ; fill in operation
                     03E9  1209
                     03E9  1210 ; Write indivdual mapping entries
                     03E9  1211 ;
```

```
                     03E9  1212
    54   30 A9   D0 03E9  1213 30$:    MOVL    IFB$L_JNLBDB(R9),R4                     ; restore BDB addr
      7E   53   7D 03ED  1214          MOVQ    R3,-(SP)                                ; make type and BDB args
                03F0  1215          RMSSUC                                          ; success if no jnling
09 0A A7   01   E1 03F3  1216          BBC     #RJB$V_BI,RJB$W_FLAGS(R7),40$           ; branch if no BI
      6E   02   9A 03F8  1217          MOVZBL  #CJF$_BI,(SP)                           ; set BI
         004D   30 03FB  1218          BSBW    RM$WRTJNL                               ; write the record
        26 50   E9 03FE  1219          BLBC    R0,60$                                  ; get out on error
                0401  1220
09 0A A7   02   E1 0401  1221 40$:    BBC     #RJB$V_AI,RJB$W_FLAGS(R7),50$           ; branch if no AI
      6E   03   9A 0406  1222          MOVZBL  #CJF$_AI,(SP)                           ; set AI
         003F   30 0409  1223          BSBW    RM$WRTJNL                               ; write the record
        18 50   E9 040C  1224          BLBC    R0,60$                                  ; get out on error
                040F  1225
13 0A A7   03   E1 040F  1226 50$:    BBC     #RJB$V_AT,RJB$W_FLAGS(R7),60$           ; branch if no AT
      6E   04   9A 0414  1227          MOVZBL  #CJF$_AT,(SP)                           ; set AT
   2C A9   56   D0 0417  1228          MOVL    R6,IFB$L_ATJNLBUF(R9)                   ; shortcut RJR addr.
         04D9   30 041B  1229          BSBW    COMMON_FILE_AT                          ; fill in fields
           2B   10 041E  1230          BSBB    RM$WRTJNL                               ; write the record
   52   2C A9   D0 0420  1231          MOVL    IFB$L_ATJNLBUF(R9),R2                   ; get RJR address
                0424  1232          ASSUME  RJR$L_AT_STV    EQ      RJR$L_AT_STS+4
        24 A2   7C 0424  1233          CLRQ    RJR$L_AT_STS(R2)                        ; init status
                0427  1234
      5E   08   C0 0427  1235 60$:    ADDL2   #8,SP                                   ; clear arglist
           0D   11 042A  1236          BRB     80$                                     ; exit
                042C  1237
                042C  1238 ;+
                042C  1239 ; RU mapping entry.
                042C  1240 ;-
                042C  1241
05 A6   11   90 042C  1242 70$:    MOVB    #RJR$_OPEN,RJR$B_OPER(R6)               ; fill in operation
      54   DD 0430  1243          PUSHL   R4                                      ; BDB addr
      01   DD 0432  1244          PUSHL   #CJF$_RU                                ; Set RU
      0C   10 0434  1245          BSBB    RM$WRTJNL_OBJ                           ; write the record
   5E 08   C0 0436  1246          ADDL2   #8,SP                                   ; delete arglist
                0439  1247
   5A 8E   D0 0439  1248 80$:    MOVL    (SP)+,R10                               ; restore FWA addr
   56 8E   7D 043C  1249          MOVQ    (SP)+,R6                                ; restore R6,R7
      8E   D5 043F  1250          TSTL    (SP)+                                   ; clear off call code
           05 0441  1251          RSB
```

```
                     .SBTTL  RMS$WRTJNL - Write Journal Entry
                     .SBTTL  RMS$WRTJNL_OBJ - Write Journal Entry with OBJECT_ID Flag

                     ;++
                     ; RMS$WRTJNL - Write Journal Entry
                     ; RMS$WRTJNL_OBJ - Write Journal Entry with OBJECT_ID Flag
                     ;
                     ; This routine fills in the mapping enry sequence number into the
                     ; journaling buffer and then writes it out for either a fab or rab
                     ; operation.
                     ;
                     ; Calling sequence:
                     ;
                     ;       BSBW    RMS$WRTJNL
                     ;       BSBW    RMS$WRTJNL_OBJ
                     ;
                     ; Input Parameters:
                     ;
                     ;       4(SP)   Type of journal to be written (CJF$_xx)
                     ;       8(SP)   Address of journaling BDB
                     ;       R4      Address of BDB of Related buffer
                     ;       R9      Address of IFB or IRB (depending on call)
                     ;       R10     Address of IFB if IRAB call
                     ;       R11     Address of impure area
                     ;
                     ; Implicit Inputs:
                     ;
                     ;       IFB$L_RJB       Address of RJB
                     ;       RJB$Q_CHAN      One word is used as channel for QIO
                     ;
                     ; Output Parameters:
                     ;
                     ;       R1              Destroyed
                     ;
                     ; Implicit Outputs:
                     ;
                     ;       BDB$T_JNLSEQ    One longword contains new high water mark
                     ;
                     ; Completion Codes:
                     ;
                     ;       CJF     -       CJF error, CJF status in STV
                     ;
                     ; Side Effects:
                     ;       May have switched to EXEC AST level.
                     ;
                     ;--

00000008             RBDB=8          ; stack offset to related BDB address
0000001C             JTYP=28         ; stack offset to journal type code
00000020             JBDB=32         ; stack offset to journal BDB

                     ;
                     ; Alternate Entry Point to write entry with OBJECT_ID flag.
                     ;
                     RMS$WRTJNL_OBJ::
00FC 8F    BB        PUSHR   #^M<R2,R3,R4,R5,R6,R7>                  ; save regs
53   08    D0        MOVL    #WRFLG$M_OBJECT_ID,R3                   ; set P6 flags
```

```
                07   11  0449  1310          BRB     WRTJNL
                                 044B  1311
                                 044B  1312 RM$WRTJNL::
            00FC 8F    BB  044B  1313          PUSHR   #^M<R2,R3,R4,R5,R6,R7>                   ; save regs
            53   10    D0  044F  1314          MOVL    #WRFLG$M_LOCK,R3                        ; set P6 flags
       52   1C AE      D0  0452  1315 WRTJNL: MOVL    JTYP(SP),R2                             ; get typ code
       0A   08 A9      91  0456  1316          CMPB    IRB$B_BID(R9),#IRB$C_BID                ; IRB operation?
                11      13  045A  1317          BEQL    10$                                     ; branch if yes
                                 045C  1318
                                 045C  1319 ;
                                 045C  1320 ; IFAB operation
                                 045C  1321 ;
       54   38 A9      D0  045C  1322          MOVL    IFB$L_FWA_PTR(R9),R4                    ; get FWA address
       56   00A4 C9    D0  0460  1323          MOVL    IFB$L_RJB(R9),R6                        ; get RJB address
 11 00A2 C9     02      E0  0465  1324          BBS     #IFB$V_RUP,IFB$B_JNLFLG2(R9),15$        ; branch if RUP
                1C      11  046B  1325          BRB     20$
                                 046D  1326
                                 046D  1327 ;
                                 046D  1328 ; IRAB operation
                                 046D  1329 ;
       54   38 AA      D0  046D  1330 10$:    MOVL    IFB$L_FWA_PTR(R10),R4
       56   00A4 CA    D0  0471  1331          MOVL    IFB$L_RJB(R10),R6
 0D 00A2 CA     02      E1  0476  1332          BBC     #IFB$V_RUP,IFB$B_JNLFLG2(R10),20$       ; branch if no RUP
                                 047C  1333
                                 047C  1334 ;
                                 047C  1335 ; IFB, IRB rejoin here if RU in progress.
                                 047C  1336 ;
                                 047C  1337 15$:    SSB     #WRFLG$V_RUALSO,R3                      ; set RUALSO in P6 flags
            01   52    D1  0480  1338          CMPL    R2,#CJF$_RU                             ; see if RU write
                 04    12  0483  1339          BNEQ    20$                                     ; branch if not
                                 0485  1340          SSB     #WRFLG$V_BI,R3                          ; set RU/BI in P6 flags
                                 0489  1341
                                 0489  1342 ;
                                 0489  1343 ; IFB, IRB rejoin here in no RU in progress
                                 0489  1344 ;
            55   20 AE      D0  0489  1345 20$:    MOVL    JBDB(SP),R5                             ; get jBDB address
                                 048D  1346          SSB     #BDB$V_IOP,BDB$B_FLGS(R5)               ; indicate IO in prog
            51   18 A5      D0  0492  1347          MOVL    BDB$L_ADDR(R5),R1                       ; get buff address
            02 A1    02      90  0496  1348          MOVB    #RJR$C_MAXVER,RJR$B_VERSION(R1)         ; set journal rec ver #
                     3E      BB  049A  1349          PUSHR   #^M<R1,R2,R3,R4,R5>
 08 A1   0920 C4     1C      28  049C  1350          MOVC3   #FWA$S_JNLID,FWA$T_JNLID(R4),RJR$T_JNLID(R1) ; copy journal id
                     3E      BA  04A3  1351          POPR    #^M<R1,R2,R3,R4,R5>
            57   14 A5      3C  04A5  1352          MOVZWL  BDB$W_NUMB(R5),R7                       ; get record length
         00000000'EF        16  04A9  1353          JSB     RM$SETEFN                               ; get EFN
                 01      BA  04AF  1354          POPR    #^M<R0>
                                 04B1  1355          $QIO_S -                                        ; issue QIO
                                 04B1  1356                  EFN     =       R0, -
                                 04B1  1357                  CHAN    =       RJB$Q_CHAN-2(R6)[R2], -
                                 04B1  1358                  FUNC    =       #IO$_WRITEVBLK, -
                                 04B1  1359                  IOSB    =       BDB$C_IOSB(R5), -
                                 04B1  1360                  ASTADR  =       RM$STALLAST, -
                                 04B1  1361                  ASTPRM  =       R9, -                   ; IRB/IFB
                                 04B1  1362                  P1      =       (R1), -                 ; buffer address
                                 04B1  1363                  P2      =       R7, -                   ; size of transfer
                                 04B1  1364                  P6      =       R3                      ; journal type
            18 50    E9      04D7  1365          BLBC    R0,30$                                  ; get out on error
                             04DA  1366
```

```
                00000000'EF   16  04DA  1367          JSB     RM$STALL                                ; wait for completion
                  50    48 A5 D0  04E0  1368          MOVL    BDB$L_IOSB(R5),R0                       ; retrieve status
                  52    1C AE D0  04E4  1369          MOVL    JTYP(SP),R2                             ; get typ code
                  54    08 AE D0  04E8  1370          MOVL    RBDB(SP),R4                             ; get related BDB addr
              34 A442   4C A5 D0  04EC  1371          MOVL    BDB$L_IOSB+4(R5),BDB$T_JNLSEQ-4(R4)[R2] ; retrieve seq #
                                  04F2  1372 30$:     CSB     #BDB$V_IOP,BDB$B_FLGS(R5)               ; clear IO in prog
                      00FC 8F   BA  04F7  1373          POPR    #^M<R2,R3,R4,R5,R6,R7>                  ; restore regs
                         0B 50  E8  04FB  1374          BLBS    R0,40$                                  ; get out on success
                00000000'EF   16  04FE  1375          JSB     RM$MAPERR                               ; fill in STV
                                  0504  1376          RMSERR  CJF                                     ; force CJF error
                                  0509  1377 40$:     RSB                                             ; return to caller
```

```
                     050A  1379          .SBTTL  RM$FRCJNL - Force All Journal Entries for a buffer
                     050A  1380 ;++
                     050A  1381 ; FORCE_JNL - Force Journal Entries
                     050A  1382 ;
                     050A  1383 ; This routine performs a force operation to all open journals
                     050A  1384 ; at the high water mark for a buffer.
                     050A  1385 ;
                     050A  1386 ; Calling sequence:
                     050A  1387 ;
                     050A  1388 ;       BSBW    RM$FRCJNL
                     050A  1389 ;
                     050A  1390 ; Input Parameters:
                     050A  1391 ;
                     050A  1392 ;       R4      Address of BDB of Related buffer or
                     050A  1393 ;               Zero to flush all Entries.
                     050A  1394 ;       R9      IFAB or IRAB address
                     050A  1395 ;       R10     IFAB address if IFAB operation
                     050A  1396 ;       R11     Address of Impure Area
                     050A  1397 ;
                     050A  1398 ; Implicit Inputs:
                     050A  1399 ;
                     050A  1400 ;       IFB$L_RJB       Address of RJB
                     050A  1401 ;
                     050A  1402 ; Output Parameters:
                     050A  1403 ;
                     050A  1404 ;       R1 - R3, R5     Destroyed
                     050A  1405 ;
                     050A  1406 ; Implicit Outputs:
                     050A  1407 ;       None.
                     050A  1408 ;
                     050A  1409 ; Completion Codes:
                     050A  1410 ;
                     050A  1411 ;       CJF - CJF error, Status from QIO in STV
                     050A  1412 ;
                     050A  1413 ; Side Effects:
                     050A  1414 ;       May have switched to EXEC AST level.
                     050A  1415 ;--
                     050A  1416
                     050A  1417 RM$FRCJNL::
      7E    01    D0 050A  1418          MOVL    #1,-(SP)                        ; anticipate success
 0A    08 A9    91 050D  1419          CMPB    IRB$B_BID(R9),#IRB$C_BID        ; IRB operation?
            07    13 0511  1420          BEQL    10$                             ; branch if yes
 55    00A4 C9    D0 0513  1421          MOVL    IFB$L_RJB(R9),R5                ; get RJB address
            05    11 0518  1422          BRB     15$
 55    00A4 CA    D0 051A  1423 10$:     MOVL    IFB$L_RJB(R10),R5
                     051F  1424
0C 0A A5    01    E1 051F  1425 15$:     BBC     #RJB$V_BI,RJB$W_FLAGS(R5),20$   ; branch if no BI
      52    02    D0 0524  1426          MOVL    #CJF$_BI,R2                     ; indicate BI
         005A    30 0527  1427          BSBW    FORCE_JNL                       ; go do force
        03 50    E8 052A  1428          BLBS    R0,20$                          ; skip on success
      6E    50    D0 052D  1429          MOVL    R0,(SP)                         ; save error code
                     0530  1430
0C 0A A5    02    E1 0530  1431 20$:     BBC     #RJB$V_AI,RJB$W_FLAGS(R5),30$   ; branch if no AI
      52    03    D0 0535  1432          MOVL    #CJF$_AI,R2                     ; indicate AI
         0049    30 0538  1433          BSBW    FORCE_JNL                       ; go do force
        03 50    E8 053B  1434          BLBS    R0,30$                          ; skip on success
      6E    50    D0 053E  1435          MOVL    R0,(SP)                         ; save error code
```

```
                     0541  1436
 0C 0A A5   03   E1  0541  1437 30$:    BBC     #RJB$V_AT,RJB$W_FLAGS(R5),40$           ; branch if no AT
       52   04   D0  0546  1438          MOVL    #CJF$_AT,R2                             ; indicate AT
              0038  30  0549  1439          BSBW    FORCE_JNL                               ; go do force
            03 50   E8  054C  1440          BLBS    R0,40$                                  ; skip on success
       6E   50   D0  054F  1441          MOVL    R0,(SP)                                 ; save error code
                     0552  1442
 1A 0A A5   00   E1  0552  1443 40$:    BBC     #RJB$V_RU,RJB$W_FLAGS(R5),50$           ; branch if no RU
51  00000000'9F   D0  0557  1444          MOVL    @#CTL$GL_RUF,R1                         ; RU in prog?
              11   13  055E  1445          BEQL    50$                                     ; branch if not
 0C 11 A1   01   E1  0560  1446          BBC     #RUCB$V_ACTIVE,RUCB$B_CTRL(R1),50$
       52   01   D0  0565  1447          MOVL    #CJF$_RU,R2                             ; indicate RU
              0019  30  0568  1448          BSBW    FORCE_JNL                               ; go do force
            03 50   E8  056B  1449          BLBS    R0,50$                                  ; skip on success
       6E   50   D0  056E  1450          MOVL    R0,(SP)                                 ; save error code
                     0571  1451
       50   8E   D0  0571  1452 50$:    MOVL    (SP)+,R0                                ; get worst status
         01 50   E9  0574  1453          BLBC    R0,60$                                  ; get out on success
                     05  0577  1454          RSB
  00000000'EF    16  0578  1455 60$:    JSB     RM$MAPERR                               ; fill in STV
                     057E  1456          RMSERR  CJF                                     ; force CJF error
                     05  0583  1457          RSB
```

```
0584  1459          .SBTTL  FORCE_JNL - Force Journal Entries
0584  1460
0584  1461 ;++
0584  1462 ; FORCE_JNL - Force Journal Entries
0584  1463 ;
0584  1464 ; This routine performs a force operation to the specified journal
0584  1465 ; at the high water mark for a buffer.
0584  1466 ;
0584  1467 ; Calling sequence:
0584  1468 ;
0584  1469 ;       BSBW    RM$FRCJNL
0584  1470 ;
0584  1471 ; Input Parameters:
0584  1472 ;
0584  1473 ;       R2      Type of journal to be forced (CJF$_xx)
0584  1474 ;       R4      Address of BDB of Related buffer or
0584  1475 ;               Zero to flush all entries.
0584  1476 ;       R5      Adddress of RJB
0584  1477 ;       R9      IFAB or IRAB address
0584  1478 ;       R10     IFAB address if IFAB operation
0584  1479 ;       R11     Address of Impure Area
0584  1480 ;
0584  1481 ; Implicit Inputs:
0584  1482 ;
0584  1483 ;       IFB$L_RJB       Address of RJB
0584  1484 ;       RJB$Q_CHAN      One word is used as channel for QIO
0584  1485 ;       BDB$T_JNLSEQ    One longword contains high water mark for force
0584  1486 ;
0584  1487 ; Output Parameters:
0584  1488 ;
0584  1489 ;       R0 - R3         Destroyed
0584  1490 ;
0584  1491 ; Implicit Outputs:
0584  1492 ;       None.
0584  1493 ;
0584  1494 ; Completion Codes:
0584  1495 ;
0584  1496 ;       Any QIO status value,
0584  1497 ;       Any IOSB status vaule from a journaling QIO.
0584  1498 ;
0584  1499 ; Side Effects:
0584  1500 ;       May have switched to EXEC AST level.
0584  1501 ;--
0584  1502
0584  1503 FORCE_JNL:
      50   01   D0  0584  1504          MOVL    #1,R0                                   ; anticipate success
      53   54   D0  0587  1505          MOVL    R4,R3                                   ; see if buffer present
           07   13  058A  1506          BEQL    10$                                     ; branch if not
53    34 A442   D0  058C  1507          MOVL    BDB$T_JNLSEQ-4(R4)[R2],R3               ; get high water mark
           39   13  0591  1508          BEQL    20$                                     ; if zero, bdb has not
                    0593  1509                                                          ; been used as part of a
                    0593  1510                                                          ; journaling operation.
 00000000'EF   16  0593  1511 10$:     JSB     RM$SETEFN                               ; get EFN
           01   BA  0599  1512          POPR    #^M<R0>
                    059B  1513          $QIO_S -                                        ; issue QIO
                    059B  1514                  EFN     =       R0, -
                    059B  1515                  CHAN    =       RJB$Q_CHAN-2(R5)[R2], -
```

```
                     059B  1516                    FUNC    =       #IO$_FORCE, -
                     059B  1517                    IOSB    =       IRB$L_IOS(R9), -
                     059B  1518                    ASTADR  =       RM$STALLAST, -
                     059B  1519                    ASTPRM  =       R9, -
                     059B  1520                    P2      =       R3
        0A 50   E9   05BF  1521            BLBC    R0,20$                          ; high water mark
                                                                                   ; get out on error
00000000'EF     16   05C2  1522            JSB     RM$STALL                        ; wait for completion
 50     0C A9   D0   05C8  1523            MOVL    IRB$L_IOS(R9),R0                ; retrieve status
                     05CC  1524
                 05  05CC  1525 20$:      RSB                                     ; return to caller
```

```
                     05CD  1527          .SBTTL  RMS$DSCJNL - Disconnect IRAB Journal Structures
                     05CD  1528
                     05CD  1529 ;++
                     05CD  1530 ; RMS$DSCJNL - Disconnect IRAB Journal Structures
                     05CD  1531 ;
                     05CD  1532 ; This routine deallocates the data structures for journaling record
                     05CD  1533 ; processing operations from the IRAB.
                     05CD  1534 ;
                     05CD  1535 ; Calling sequence:
                     05CD  1536 ;
                     05CD  1537 ;       BSBW    RMS$DSCJNL
                     05CD  1538 ;
                     05CD  1539 ; Input Parameters:
                     05CD  1540 ;
                     05CD  1541 ;       R9      Address of IRAB
                     05CD  1542 ;       R11     Address of Impure area
                     05CD  1543 ;
                     05CD  1544 ; Implicit Inputs:
                     05CD  1545 ;
                     05CD  1546 ;       IRB$L_JNLBDB    Address of journaling BDB
                     05CD  1547 ;
                     05CD  1548 ; Output Parameters:
                     05CD  1549 ;       R0 - R5         Destroyed
                     05CD  1550 ;
                     05CD  1551 ; Implicit Outputs:
                     05CD  1552 ;       None.
                     05CD  1553 ;
                     05CD  1554 ; Completion Codes:
                     05CD  1555 ;       None.
                     05CD  1556 ;
                     05CD  1557 ; Side Effects:
                     05CD  1558 ;       None.
                     05CD  1559 ;
                     05CD  1560 ;--
                     05CD  1561
                     05CD  1562 RMS$DSCJNL::
                     05CD  1563
 54    30 A9    D0    05CD  1564          MOVL    IRB$L_JNLBDB(R9),R4             ; get journal BDB address
       09       13    05D1  1565          BEQL    10$                             ; skip if none
00000000'EF       16    05D3  1566          JSB     RMS$RETJNLBDB                   ; deallocate it
       30 A9    D4    05D9  1567          CLRL    IRB$L_JNLBDB(R9)                ; clear pointer
                     05DC  1568 10$:                                              ;
 54    2C A9    D0    05DC  1569          MOVL    IRB$L_ATJNLFLF(R9),R4           ; get AT MJB address
       0F       13    05E0  1570          BEQL    20$                             ; branch if none
    55   54    D0    05E2  1571          MOVL    R4,R5                           ; copy MJB address for FORCE call
      0188    30    05E5  1572          BSBW    RMS$FORCE_MJB                   ; force the IRB AT journaling record
                     05E8  1573                                                  ; Note, errors eaten!
00000000'EF       16    05E8  1574          JSB     RMS$RETBLK1                     ; give it up
       2C A9    D4    05EE  1575          CLRL    IRB$L_ATJNLBUF(R9)              ; clear pointer
                05    05F1  1576 20$:     RSB
```

```
                     05F2  1578          .SBTTL  RMSDEAJNL - Close journaling on file
                     05F2  1579
                     05F2  1580 ;++
                     05F2  1581 ; RMSDEAJNL - Close journaling on file
                     05F2  1582 ;
                     05F2  1583 ; This routine deassigns the journal channels open for the file and
                     05F2  1584 ; deallocates the journaling data structures from the IFAB.
                     05F2  1585 ;
                     05F2  1586 ; Calling sequence:
                     05F2  1587 ;
                     05F2  1588 ;       BSBW    RMSDEAJNL
                     05F2  1589 ;
                     05F2  1590 ; Input Parameters:
                     05F2  1591 ;
                     05F2  1592 ;       R9      Address of IFAB
                     05F2  1593 ;       R11     Impure area address
                     05F2  1594 ;
                     05F2  1595 ; Implicit Inputs:
                     05F2  1596 ;
                     05F2  1597 ;       IRBSL_RJB       Address of RJB
                     05F2  1598 ;
                     05F2  1599 ; Output Parameters:
                     05F2  1600 ;
                     05F2  1601 ;       R1 - R5         Destroyed
                     05F2  1602 ;
                     05F2  1603 ; Implicit Outputs:
                     05F2  1604 ;       None.
                     05F2  1605 ;
                     05F2  1606 ; Completion Codes:
                     05F2  1607 ;       CJF     - CJF Operation Error, CJF status from $DEASJNL in STV
                     05F2  1608 ;
                     05F2  1609 ; Side Effects:
                     05F2  1610 ;       None.
                     05F2  1611 ;
                     05F2  1612 ;--
                     05F2  1613
                     05F2  1614 RMSDEAJNL::
                     05F2  1615
     7E    01    D0  05F2  1616          MOVL    #1,-(SP)                        ; assume success
   54    30 A9    D0  05F5  1617          MOVL    IFBSL_JNLBDB(R9),R4             ; jnl BDB/Buffer address
         11    13  05F9  1618          BEQL    2$                              ; skip if none
         5A    DD  05FB  1619          PUSHL   R10                             ; save R10
   5A    59    D0  05FD  1620          MOVL    R9,R10                          ; R10 must be IFAB
00000000'EF    16  0600  1621          JSB     RMSRETJNLBDB                    ; deallocate BDB/Buffer
   5A    8E    D0  0606  1622          MOVL    (SP)+,R10                       ; restore R10
      30 A9    D4  0609  1623          CLRL    IFBSL_JNLBDB(R9)                ; clear pointer
      2C A9    D4  060C  1624 2$:      CLRL    IFBSL_ATJNLBUF(R9)              ; clear shortcut pointer
                   060F  1625                                                  ; to AT RJR
   54    34 A9    D0  060F  1626          MOVL    IFBSL_EXTJNLBUF(R9),R4          ; get extend MJB address
         09    13  0613  1627          BEQL    5$                              ; branch if none
00000000'EF    16  0615  1628          JSB     RMSRETBLK1                      ; give it up
      34 A9    D4  061B  1629          CLRL    IFBSL_EXTJNLBUF(R9)             ; clear pointer
   54   00A4 C9    D0  061E  1630 5$:      MOVL    IFBSL_RJB(R9),R4                ; get RJB address
         03    12  0623  1631          BNEQ    7$                              ; skip if none
      006F    31  0625  1632          BRW     45$                             ; get out
13 0A A4    01    E5  0628  1633 7$:      BBCC    #RJBSV_BI,RJBSW_FLAGS(R4),10$   ; branch if no BI
                   062D  1634          $DEASJNL_S -
```

```
                     062D  1635                       CHAN = RJB$W_BICHAN(R4)
                     063A  1636
          03 50      E8  063A  1637              BLBS    R0,10$                          ; continue on success
        6E   50      D0  063D  1638              MOVL    R0,(SP)                         ; save error code
                     0640  1639
13 0A A4    02      E5  0640  1640 10$:         BBCC    #RJB$V_AI,RJB$W_FLAGS(R4),20$   ; branch if no AI
                     0645  1641                 $DEASJNL_S -                            ; deassign channel
                     0645  1642                         CHAN = RJB$W_AICHAN(R4)
          03 50      E8  0652  1643              BLBS    R0,20$                          ; continue on success
        6E   50      D0  0655  1644              MOVL    R0,(SP)                         ; save error code
                     0658  1645
13 0A A4    03      E5  0658  1646 20$:         BBCC    #RJB$V_AT,RJB$W_FLAGS(R4),30$   ; branch if no AT
                     065D  1647                 $DEASJNL_S -                            ; deassign channel
                     065D  1648                         CHAN = RJB$W_ATCHAN(R4)
          03 50      E8  066A  1649              BLBS    R0,30$                          ; continue on success
        6E   50      D0  066D  1650              MOVL    R0,(SP)                         ; save error code
                     0670  1651
12 0A A4    00      E5  0670  1652 30$:         BBCC    #RJB$V_RU,RJB$W_FLAGS(R4),40$   ; branch if no RU
                     0675  1653                 $DEASJNL_S -                            ; deassign channel
                     0675  1654                         CHAN = RJB$W_RUCHAN(R4)
          03 50      E8  0681  1655              BLBS    R0,40$                          ; continue on success
        6E   50      D0  0684  1656              MOVL    R0,(SP)                         ; save error code
                     0687  1657
          0A A4      B4  0687  1658 40$:         CLRW    RJB$W_FLAGS(R4)                 ; clear open flags
        53   59      D0  068A  1659              MOVL    R9,R3                           ; deallocate RJB
  00000000'EF        16  068D  1660              JSB     RM$RETBLK                       ;
      00A4 C9        D4  0693  1661              CLRL    IFB$L_RJB(R9)                   ; evaporate pointer
        50   8E      D0  0697  1662 45$:         MOVL    (SP)+,R0                        ; get true error code
          01 50      E9  069A  1663              BLBC    R0,50$                          ; get out on error
                     05  069D  1664              RSB
                     069E  1665
00000000'EF          16  069E  1666 50$:         JSB     RM$MAPERR                       ; set STV
                     06A4  1667                  RMSERR  CJF                             ; force CJF error
                     05  06A9  1668              RSB                                     ; return to caller
```

```
                        06AA  1670          .SBTTL  RMSALLOC_MJB - Alloc and init MJB
                        06AA  1671
                        06AA  1672 ;++
                        06AA  1673 ;
                        06AA  1674 ; RMSALLOC_MJB - allocate and initalize a miscellaneous journaling buffer
                        06AA  1675 ;
                        06AA  1676 ;       The MJB is used for audit trail entries and AI extend descriptions.
                        06AA  1677 ;
                        06AA  1678 ; Calling Sequence:
                        06AA  1679 ;
                        06AA  1680 ;       BSBW    RMSALLOC_MJB
                        06AA  1681 ;
                        06AA  1682 ; Input Parameters:
                        06AA  1683 ;
                        06AA  1684 ;       R10     IFAB address
                        06AA  1685 ;       R2      mjb size in bytes
                        06AA  1686 ;
                        06AA  1687 ; Output Parameters:
                        06AA  1688 ;
                        06AA  1689 ;       R0      status
                        06AA  1690 ;       R1      MJB address
                        06AA  1691 ;
                        06AA  1692 ; Side Effects, Implicit Inputs, Implicit Outputs:
                        06AA  1693 ;
                        06AA  1694 ;       None.
                        06AA  1695 ;
                        06AA  1696 ;--
                        06AA  1697
                        06AA  1698 RMSALLOC_MJB::
                        06AA  1699
                        06AA  1700          ASSUME  <IRB$C_BID&1>   EQ      0
                        06AA  1701          ASSUME  <IFB$C_BID&1>   EQ      1
                        06AA  1702          ASSUME  IFB$B_BID       EQ      IRB$B_BID
                        06AA  1703
          51    59  D0  06AA  1704          MOVL    R9,R1                           ; assume ifab addr in r1
        03 08 A9  E8  06AD  1705          BLBS    IFB$B_BID(R9),5$                ; branch if structure is ifab
          51    69  D0  06B1  1706          MOVL    IRB$L_IFAB_LNK(R9),R1           ; get ifab address from irab
                        06B4  1707 5$:
          52    07  C0  06B4  1708          ADDL2   #7,R2                           ; round request up
          52    07  CA  06B7  1709          BICL2   #7,R2                           ; ...
    52    52    FE  8F  78  06BA  1710          ASHL    #-2,R2,R2                       ; change bytes to longwords
        00000000'EF  16  06BF  1711          JSB     RMSGETBLK                       ; alloc an MJB on IFB page
              1A  50  E9  06C5  1712          BLBC    R0,10$                          ; get out on error
        08 A1     18  90  06C8  1713          MOVB    #MJB$C_BID,MJB$B_BID(R1)        ; identify MJB as MJB
      14 A1    20  A1  DE  06CC  1714          MOVAL   MJB$T_RJR(R1),MJB$L_POINTER(R1) ; init descriptor
                3E  BB  06D1  1715          PUSHR   #^M<R1,R2,R3,R4,R5>             ; save MOVC5 regs
          51    20  A1  DE  06D3  1716          MOVAL   MJB$T_RJR(R1),R1                ; get RJR address
61  38  00    61    00  2C  06D7  1717          MOVC5   #0,(R1),#0,#RJR$C_HDRLEN,(R1)   ; zero the RJR overhead
                3E  BA  06DD  1718          POPR    #^M<R1,R2,R3,R4,R5>             ; restore MOVC5 regs
                        06DF  1719          RMSSUC
                    05  06E2  1720 10$:     RSB                                     ; return to caller
```

```
                     06E3  1722          .SBTTL  RMSWRITE_MJB - Write Miscellaneous Journaling Buffer
                     06E3  1723
                     06E3  1724 ;++
                     06E3  1725 ;
                     06E3  1726 ; RMSWRITE_MJB
                     06E3  1727 ;
                     06E3  1728 ; This routine is used to write a journaling record described by a
                     06E3  1729 ; miscellaneous journaling buffer.
                     06E3  1730 ;
                     06E3  1731 ; Calling Sequence:
                     06E3  1732 ;
                     06E3  1733 ;       BSBW    RMSWRITE_MJB
                     06E3  1734 ;
                     06E3  1735 ; Input Parameters:
                     06E3  1736 ;
                     06E3  1737 ;       R9      - IFAB or IRAB address
                     06E3  1738 ;       R5      - address of MJB
                     06E3  1739 ;
                     06E3  1740 ; Implicit Inputs:
                     06E3  1741 ;
                     06E3  1742 ;       MJB fields:
                     06E3  1743 ;
                     06E3  1744 ;               JNL     - CJF$_AI, BI, AT, or RU for journal channel to use
                     06E3  1745 ;               FLAGS   - various
                     06E3  1746 ;               DESC    - descriptor of embedded RJR to write
                     06E3  1747 ;
                     06E3  1748 ; Output Parameters:
                     06E3  1749 ;
                     06E3  1750 ;       R0      - status
                     06E3  1751 ;       R6      - destroyed
                     06E3  1752 ;
                     06E3  1753 ; Implicit Outputs:
                     06E3  1754 ;
                     06E3  1755 ;       MJB IOSB has status of operation.
                     06E3  1756 ;
                     06E3  1757 ; Side Effects:
                     06E3  1758 ;
                     06E3  1759 ;       None.
                     06E3  1760 ;
                     06E3  1761 ;--
                     06E3  1762
                     06E3  1763 RMSWRITE_MJB::
                     06E3  1764
           1C    BB  06E3  1765          PUSHR   #^M<R2,R3,R4>                   ; save work registers
      54   59    D0  06E5  1766          MOVL    R9,R4                           ; get potential IFAB address
                     06E8  1767
                     06E8  1768          ASSUME  IFB$B_BID       EQ      IRB$B_BID
                     06E8  1769
     0B    08 A4    91  06E8  1770          CMPB    IFB$B_BID(R4),#IFB$C_BID        ; file or record operation?
           03    13  06EC  1771          BEQL    5$                              ; branch if IFAB
      54   69    D0  06EE  1772          MOVL    IRB$L_IFAB_LNK(R9),R4           ; get IFAB address
                     06F1  1773
  56   00A4 C4    D0  06F1  1774 5$:     MOVL    IFB$L_RJB(R4),R6                ; get pointer to RJB
           5D    13  06F6  1775          BEQL    35$                             ; branch if none
                     06F8  1776
52 0A A5    00    E1  06F8  1777          BBC     #MJB$V_INIT,MJB$W_FLAGS(R5),35$ ; skip if RJR in MJB is useless
                     06FD  1778
```

```
                53   D4  06FD  1779          CLRL    R3                              ; initialize MODIFIER flags
04 0A A5        01   E1  06FF  1780          BBC     #MJB$V_FORCE,MJB$W_FLAGS(R5),10$; skip if not write-thru to jnl
   53     4C 8F    90  0704  1781          MOVB    #WRMOD$M_FORCE,R3               ; indicate write-thru to jnl
                       0708  1782
   52     0C A5    9A  0708  1783 10$:     MOVZBL  MJB$B_JNL(R5),R2                ; get JNL type for channel calculati
          54       59  D0  070C  1784          MOVL    R9,R4                           ; initialize astparm to IRAB address
03 0A A5        02   E1  070F  1785          BBC     #MJB$V_FILE,MJB$W_FLAGS(R5),20$ ; branch if assumption OK
          54       5A  D0  0714  1786          MOVL    R10,R4                          ; otherwise astprm is IFAB address
                       0717  1787
  00000000'EF   16   0717  1788 20$:     JSB     RM$SETEFN                       ; get an EFN to wait on
                01   BA  071D  1789          POPR    #^M<R0>                         ; and stick it in R0
                       071F  1790
                       071F  1791          $WRITEJNL_S -
                       071F  1792                  CHAN    = RJB$Q_CHAN-2(R6)[R2], -       ; channel of journal
                       071F  1793                  WRTBUF  = MJB$Q_DESC(R5), -             ; RJR descriptor
                       071F  1794                  MODIF   = R3,-                          ; modifier flags
                       071F  1795                  EFN     = R0,-                          ; event flag to wait on
                       071F  1796                  IOSB    = IRB$L_IOS(R9),-               ; status of operation
                       071F  1797                  ASTADR  = RM$STALLAST,-                 ; back to RM$STALLAST
                       071F  1798                  ASTPRM  = R4                            ; IFAB or IRAB
                       073F  1799
          21   50   E9  073F  1800          BLBC    R0,50$                          ; go away on error
14 0A A5        03   E0  0742  1801          BBS     #MJB$V_SYNCH_SHARE,MJB$W_FLAGS(R5),40$  ; branch if SFSB lock
                       0747  1802                                                  ; can't be given up
  00000000'EF   16   0747  1803          JSB     RM$STALL                        ; wait for completion
18 A5     0C A9    7D  074D  1804 30$:     MOVQ    IRB$L_IOS(R9),MJB$Q_IOSB(R5)    ; save status and seq no in MJB
          0E   50   E9  0752  1805          BLBC    R0,50$                          ; go away on error
                       0755  1806 35$:
                1C   BA  0755  1807          POPR    #^M<R2,R3,R4>                   ; restore registers
                       0757  1808          RMSSUC                                  ; indicate success
                05     075A  1809          RSB                                     ; return to caller
                       075B  1810
00000000'EF     16   075B  1811 40$:     JSB     RM$STALL_LOCK                   ; wait, keeping file lock (used for
                       0761  1812                                                  ; extend)
                EA   11  0761  1813          BRB     30$                             ; go check status
                       0763  1814
                       0763  1815 50$:
                1C   BA  0763  1816          POPR    #^M<R2,R3,R4>                   ; restore work registers
                       0765  1817          RMSERR  CJF,R1                          ; default error status
00000000'EF     17   076A  1818          JMP     RM$MAPERR                       ; map error code and return
                       0770  1819                                                  ; to caller
```

```
                        0770  1821          .SUBTITLE RMSFORCE_MJB - Force MJB Entries
                        0770  1822 ;++
                        0770  1823 ; RMSFORCE_MJB
                        0770  1824 ;
                        0770  1825 ;       This routine is called at disconnect to force the journal entries
                        0770  1826 ;       described by the high water mark in the MJB.  (Currently only used
                        0770  1827 ;       for AT record operations.
                        0770  1828 ;
                        0770  1829 ; Inputs:
                        0770  1830 ;       r5      MJB address
                        0770  1831 ;
                        0770  1832 ; Implicit Inputs:
                        0770  1833 ;       contents of the MJB, including MJB$B_JNL and the sequence number
                        0770  1834 ;       in the IOSB.
                        0770  1835 ;
                        0770  1836 ;       rjb has the channel assigned to the AT journal
                        0770  1837 ;
                        0770  1838 ; Outputs:
                        0770  1839 ;       r0 - success or failure
                        0770  1840 ;
                        0770  1841 ; Side Effects:
                        0770  1842 ;
                        0770  1843 ;       AT record journal entries flushed.
                        0770  1844 ;
                        0770  1845 ;--
                        0770  1846
                        0770  1847 RMSFORCE_MJB::
                        0770  1848
                        0770  1849          RMSSUC                                  ; default to success
                  3C    BB  0773  1850          PUSHR   #^M<R2,R3,R4,R5>                ; save work registers
            54  00A4  CA    0775  1851          MOVL    IFB$L_RJB(R10),R4               ; get RJB address
                  39    13  077A  1852          BEQL    40$                             ; get out if none
              52  0C  A5    077C  1853          MOVZBL  MJB$B_JNL(R5),R2                ; get JNL identifier
        00000000'EF       16  0780  1854          JSB     RMSSETEFN                       ; allocate an event flag
                  01    BA  0786  1855          POPR    #^M<R0>                         ; get EF in R0
                        0788  1856
                        0788  1857          $FORCEJNL_S -
                        0788  1858                  CHAN    = RJB$Q_CHAN-2(R4)[R2], -       ; channel of journal
                        0788  1859                  SEQNO   = MJB$Q_IOSB+4(R5), -           ; sequence number
                        0788  1860                  EFN     = R0  -                         ; event flag
                        0788  1861                  IOSB    = IRB$L_IOS(R9), -              ; use IOSB in IRB
                        0788  1862                  ASTADR  = RMSSTALLAST, -                ; usual AST address
                        0788  1863                  ASTPRM = R9                             ; IRAB operation
                        07A4  1864
                  11    50  E9  07A4  1865          BLBC    R0,50$                          ; out on error
        00000000'EF       16  07A7  1866          JSB     RMSSTALL                        ; wait for completion
      18  A5  0C  A9        7D  07AD  1867          MOVQ    IRB$L_IOS(R9),MJB$Q_IOSB(R5)    ; grab status for fun
                  03    50  E9  07B2  1868          BLBC    R0,50$                          ; out on error
                  3C    BA  07B5  1869 40$:     POPR    #^M<R2,R3,R4,R5>                ; restore work registers
                        05  07B7  1870          RSB                                     ; return to caller
                        07B8  1871
                        07B8  1872 50$:     RMSERR  CJF,R1                          ; cjf error
        00000000'EF       16  07BD  1873          JSB     RMSMAPERR                       ; map the error code
                  F0    11  07C3  1874          BRB     40$                             ; return to caller
```

```
                                                B 3
                                07C5  1876          SUBTITLE RMS$ALLOC_RJB_BDB - Allocate RJB, Journal BDB
                                07C5  1877 ;++
                                07C5  1878 ; RMS$ALLOC_RJB_BDB
                                07C5  1879 ;
                                07C5  1880 ;       This routine allocates an RJB and JNL BDB for use by RMS journaling.
                                07C5  1881 ;
                                07C5  1882 ; Inputs:
                                07C5  1883 ;       R9      IFAB
                                07C5  1884 ;
                                07C5  1885 ; Outputs:
                                07C5  1886 ;       R0      status
                                07C5  1887 ;       IFB$L_JNLBDB    address of JNL BDB
                                07C5  1888 ;       IFB$L_RJB       address of RJB
                                07C5  1889 ;
                                07C5  1890 ; Side Effects:
                                07C5  1891 ;       None.
                                07C5  1892 ;
                                07C5  1893 ;--
                                07C5  1894
                                07C5  1895 RMS$ALLOC_RJB_BDB::
                                07C5  1896
          38   BB                07C5  1897          PUSHR   #^M<R3,R4,R5>                   ; save work registers
      00A4 C9   D5                07C7  1898          TSTL    IFB$L_RJB(R9)                   ; RJB present?
           1E   12                07CB  1899          BNEQ    10$                             ; branch if yes
      51   59   D0                07CD  1900          MOVL    R9,R1                           ; allocate RJB
      52   03   D0                07D0  1901          MOVL    #RJB$C_BLN/4,R2                 ; size of RJB
00000000'EF   16                07D3  1902          JSB     RM$GETBLK                       ; get it
        61 50   E9                07D9  1903          BLBC    R0,30$                          ; get out on error
  00A4 C9   51   D0                07DC  1904          MOVL    R1,IFB$L_RJB(R9)                ; save RJB address
    08 A1   16   90                07E1  1905          MOVB    #RJB$C_BID,RJB$B_BID(R1)        ; initialize RJB
                                07E5  1906          SSB     #IFB$V_JNL,IFB$B_JNLFLG2(R9)    ; indicate RJB present
          30 A9   D5                07EB  1907 10$:     TSTL    IFB$L_JNLBDB(R9)                ; JNLBDB already allocated?
              4A   12                07EE  1908          BNEQ    20$                             ; branch if so
                                07F0  1909
                                07F0  1910 ;
                                07F0  1911 ; If AI journaling a relative file - allocate a bigger buffer, on large enough
                                07F0  1912 ; to contain prolog (512 bytes).
                                07F0  1913 ;
 0D 00A0 C9   03   E1           07F0  1914          BBC     #IFB$V_AI,IFB$B_JNLFLG(R9),15$  ; skip if not AI journaling
       23 A9   01   91           07F6  1915          CMPB    #IFB$C_REL,IFB$B_ORGCASE(R9)    ; is it relative file?
              07   12           07FA  1916          BNEQ    15$                             ; branch if not relative
                                07FC  1917
                                07FC  1918          ASSUME  <RJR$C_BLKLEN+512> GT RJR$C_FILNAMLEN
                                07FC  1919
      55   0244 8F   3C           07FC  1920          MOVZWL  #<RJR$C_BLKLEN+512>,R5          ; size of buffer
              05   11           0801  1921          BRB     16$                             ; join common code
                                0803  1922 15$:                                             ;
      55   01C4 8F   3C           0803  1923          MOVZWL  #RJR$C_FILNAMLEN,R5             ; size of buffer to allocate
           7E   5A   D0           0808  1924 16$:     MOVL    R10,-(SP)                       ; save R10, ALDJNLBUF needs R10=IFB
           5A   59   D0           080B  1925          MOVL    R9,R10                          ; copy IFB address
55    000001FF 8F   C0           080E  1926          ADDL2   #511,R5                         ; round to page boundary
55    000001FF 8F   CA           0815  1927          BICL2   #511,R5                         ;
      00000000'EF   16           081C  1928          JSB     RM$ALDJNLBUF                    ; allocate jnl BDB and buffer
           5A   8E   D0           0822  1929          MOVL    (SP)+,R10                       ; restore R10
             18 50   E9           0825  1930          BLBC    R0,40$                          ; get out on error
        30 A9   54   D0           0828  1931          MOVL    R4,IFB$L_JNLBDB(R9)             ; save address of JNLBDB
              3E   BB           082C  1932          PUSHR   #^M<R1,R2,R3,R4,R5>             ; save regs zeroed by MOVC5
```

```
                    51   18 A4   D0  082E  1933          MOVL    BDB$L_ADDR(R4),R1                       ; get RJR address
        61   38   00   61    00   2C  0832  1934          MOVC5   #0,(R1),#0,#RJR$C_HDRLEN,(R1)           ; zero the RJR overhead
                          3E   BA  0838  1935          POPR    #^M<R1,R2,R3,R4,R5>                     ; restore regs zeroed by MOVC5
                               083A  1936
                               083A  1937 20$:      RMSSUC                                              ; success
                          38   BA  083D  1938 30$:      POPR    #^M<R3,R4,R5>                           ; restore registers
                          05   083F  1939          RSB                                                 ; to caller
                               0840  1940 40$:                                                          ; deallocate the RJB
                     7E   50   D0  0840  1941          MOVL    R0,-(SP)                                ; save error code
                     53   59   D0  0843  1942          MOVL    R9,R3                                   ; address of block holding space
                54   00A4 C9   D0  0846  1943          MOVL    IFB$L_RJB(R9),R4                        ; address of RJB
                 00000000'EF   16  084B  1944          JSB     RM$RETBLK                               ; return space and to caller
                     50   8E   D0  0851  1945          MOVL    (SP)+,R0                                ; restore error code
                          38   BA  0854  1946          POPR    #^M<R3,R4,R5>                           ; restore registers
                          05   0856  1947          RSB                                                 ; to caller
```

```
                     1949          .SUBTITLE RM$AT_JNL_RECORD - Write AT Entry for Records
                     1950
                     1951  ;++
                     1952  ; RM$AT_JNL_RECORD
                     1953  ;
                     1954  ;       This routine is responsible for writing any AT journaling record
                     1955  ;       required to describe a record operation.  This routine's primary
                     1956  ;       task is to make sure the RJR overhead is filled in properly, and
                     1957  ;       the correct MJB inputs are set.  RM$WRITE_MJB is then called to
                     1958  ;       actually perform the CJF write service.
                     1959  ;
                     1960  ; Calling Sequence:
                     1961  ;
                     1962  ;       BSBW    RM$AT_JNL_RECORD
                     1963  ;
                     1964  ;       This routine is called only by RM$EXRMS.
                     1965  ;
                     1966  ; Input Parameters:
                     1967  ;
                     1968  ;       R0      operation status to this point
                     1969  ;       R8      user's RAB
                     1970  ;       R9      IRAB
                     1971  ;       R10     IFAB
                     1972  ;
                     1973  ; Implicit Inputs:
                     1974  ;
                     1975  ;       IRB$L_ATJNLBUF  - pointer to MJB containing RJR
                     1976  ;       RJR$B_OPER      - must be filled in by caller
                     1977  ;
                     1978  ; Output Parameters:
                     1979  ;
                     1980  ;       r0      operation status
                     1981  ;       r1      destroyed
                     1982  ;
                     1983  ; Implicit Outputs:
                     1984  ;
                     1985  ;       None.   (for now)
                     1986  ;
                     1987  ; Side Effects:
                     1988  ;
                     1989  ;       RJR written to CJF
                     1990  ;
                     1991  ;--
                     1992
                     1993  RM$AT_JNL_RECORD::
                     1994
          59   D5    1995          TSTL    R9                      ; any structure address?
          01   12    1996          BNEQ    2$                      ; if no, must be structureless exit
               05    1997  1$:     RSB                             ; nothing to do
                     1998
                     1999          ASSUME  IFB$B_BID       EQ      IRB$B_BID
                     2000
    0A    08 A9  91    2001  2$:     CMPB    IFB$B_BID(R9),#IRB$C_BID ; is this an IRAB?
          F9   12    2002          BNEQ    1$                      ; if neq no, forget it
                     2003
F3  00A0  CA   04   E1  2004          BBC     #IFB$V_AT,IFB$B_JNLFLG(R10),1$ ; skip if not AT journaling
          30   BB    2005          PUSHR   #^M<R4,R5>              ; save work registers
```

```
                55   2C A9    D0  086A  2006          MOVL    IRB$L_ATJNLBUF(F9),R5   ; get MJB address
                     67  13    13  086E  2007          BEQL    70$                     ; skip if none
                                0870  2008
                                0870  2009 ;
                                0870  2010 ; Fill in required MJB fields
                                0870  2011 ;
                     0C A5  04  90  0870  2012          MOVB    #CJF$_AT,MJB$B_JNL(R5)  ; indicate we're audit trail journaling
                        0A A5   B4  0874  2013          CLRW    MJB$W_FLAGS(R5)         ; nothing special for WRITEJNL call
              10 A5   004C 8F   3C  0877  2014          MOVZWL  #RJR$C_AT_RECLEN,MJB$Q_DESC(R5) ; set up record length
                                087D  2015
                     54   20 A5  DE  087D  2016          MOVAL   MJB$T_RJR(R5),R4        ; get RJR address in R4
                          05 A4  D5  0881  2017          TSTL    RJR$B_OPER(R4)                  ; any op specified?
                             51  13  0884  2018          BEQL    70$                             ; skip if none
                4F 0A A5     00  E3  0886  2019          BBCS    #MJB$V_INIT,MJB$W_FLAGS(R5),90$ ; skip filling in RJR if already
                                088B  2020                                                      ; done
                                088B  2021 10$:                                                 ; RJR overhead filled in
                     24 A4   50  D0  088B  2022          MOVL    R0,RJR$L_AT_STS(R4)             ; get status
                                088F  2023          SSB     #16,RJR$L_AT_STS(R4)            ; make it an RMS status
                28 A4   0C A8  D0  0894  2024          MOVL    RAB$L_STV(R8),RJR$L_AT_STV(R4)  ; and get STV also
                                0899  2025 ;
                                0899  2026 ; Pull user's request from RAB into journal record.  Must probe structures.
                                0899  2027 ; All relevant data that was available at the beginning of the operation
                                0899  2028 ; is already in the journal record.  It was put there by RM$AT_COM_RAB.
                                0899  2029 ;
                             58 D5  0899  2030 20$:     TSTL    R8                              ; user structure?
                             17 13  089B  2031          BEQL    60$                             ; branch if no RAB
                                089D  2032          IFNORD  #RAB$C_BLN,(R8),60$             ; skip rest if not readable
                          01 68 91  08A5  2033          CMPB    (R8),#RAB$C_BID                 ; is it a RAB?
                             0A 12  08A8  2034          BNEQ    60$                             ; branch if no RAB
                                08AA  2035
                                08AA  2036 ;
                                08AA  2037 ; We found a readable RAB, now fill AT entry in with the RAB contents.
                                08AA  2038 ;
                   44 A4   10 A8  D0  08AA  2039          MOVL    RAB$L_RFA0(R8),RJR$L_AT_RFA0(R4);  1st part of RFA
                   48 A4   14 A8  B0  08AF  2040          MOVW    RAB$W_RFA4(R8),RJR$W_AT_RFA4(R4);  2nd part of RFA
                                08B4  2041
                      51   41 A4  9A  08B4  2042 60$:     MOVZBL  RJR$B_AT_KSZ(R4),R1             ; get key size
                      10 A5   51  C0  08B8  2043          ADDL2   R1,MJB$Q_DESC(R5)               ; account for key size
                           FE24   30  08BC  2044          BSBW    RM$WRITE_MJB                    ; write the AT record
                                08BF  2045
                                08BF  2046          ASSUME  RJR$L_AT_STV    EQ      RJR$L_AT_STS+4
                                08BF  2047
                             24 A4  7C  08BF  2048          CLRQ    RJR$L_AT_STS(R4)                ; init status for next time
                             05 A4  94  08C2  2049          CLRB    RJR$B_OPER(R4)                  ; and operation
                                08C5  2050
                                08C5  2051 ;
                                08C5  2052 ; Now zero search KEY so it doesn't linger in the buffer.
                                08C5  2053 ;
                      51   41 A4  9A  08C5  2054          MOVZBL  RJR$B_AT_KSZ(R4),R1             ; get key size for clear
                             0C  13  08C9  2055          BEQL    70$                             ; skip if none
                             0F  BB  08CB  2056          PUSHR   #^M<R0,R1,R2,R3>                ; save MOVC3 registers
4C A4   51   00   4C A4   00  2C  08CD  2057          MOVC5   #0,RJR$T_AT_KEY(R4),#0,R1,-     ; zero out KEY for next time
                                08D5  2058                  RJR$T_AT_KEY(R4)                ;
                             0F  BA  08D5  2059          POPR    #^M<R0,R1,R2,R3>                ; restore MOVC3 registers
                                08D7  2060
                             30  BA  08D7  2061 70$:     POPR    #^M<R4,R5>                      ; restore work registers
                                 05  08D9  2062 80$:     RSB                             ; return to caller
```

```
                                08DA  2063
                                08DA  2064 90$:                                 ; fill in RJR overhead
                                08DA  2065
                                08DA  2066          ASSUME RJR$B_ENTRY_TYPE EQ <RJR$B_VERSION+1>
                                08DA  2067
                      0602 8F   B0  08DA  2068          MOVW    #<<RJR$C_AT_RECORD@8>+RJR$C_MAXVER>,-
                        02 A4       08DE  2069                  RJR$B_VERSION(R4)           ; version, type
                04 A4   23 AA   90  08E0  2070          MOVB    IFB$B_ORGCASE(R10),RJR$B_ORG(R4); file organization
                           3F   BB  08E5  2071          PUSHR   #^M<R0,R1,R2,R3,R4,R5>      ; save registers MOVC3 destroys
                    55     38 AA   D0  08E7  2072          MOVL    IFB$L_FWA_PTR(R10),R5       ; get FWA address
        08 A4     0920 C5    1C   28  08EB  2073          MOVC3   #FWA$S_JNLID,FWA$T_JNLID(R5),RJR$T_JNLID(R4) ; journal id
                           3F   BA  08F2  2074          POPR    #^M<R0,R1,R2,R3,R4,R5>      ; restore MOVC3 registers
                      FF94      31  08F4  2075          BRW     10$                         ; join common code
```

```
                     08F7  2077          .SUBTITLE COMMON_FILE_AT - Get common AT file data
                     08F7  2078 ;++
                     08F7  2079 ; COMMON_FILE_AT
                     08F7  2080 ;
                     08F7  2081 ;       This routine is used to fill in the AT journal entry with data from the
                     08F7  2082 ;       IFAB at MAPJNL time.
                     08F7  2083 ;
                     08F7  2084 ; Inputs:
                     08F7  2085 ;
                     08F7  2086 ;       r8      FAB
                     08F7  2087 ;       r9      IFAB
                     08F7  2088 ;
                     08F7  2089 ; Outputs:
                     08F7  2090 ;
                     08F7  2091 ;       AT journal record fields filled in.
                     08F7  2092 ;
                     08F7  2093 ; Side Effects:
                     08F7  2094 ;
                     08F7  2095 ;       Currently, the STS/STV is forced to success due to difficulties
                     08F7  2096 ;       in acquiring the info when the journal entry must be written.
                     08F7  2097 ;       (IE,, can't do it at exit RMS like record operations because
                     08F7  2098 ;       data structures must be deallocated at release time. Better
                     08F7  2099 ;       solution is to make file AT info hendled by an MJB also, and write
                     08F7  2100 ;       and deallocate the file MJB at exit RMS.)
                     08F7  2101 ;--
                     08F7  2102
                     08F7  2103 COMMON_FILE_AT:
                     08F7  2104
        04   BB       08F7  2105          PUSHR   #^M<R2>                     ; save work register
    52  2C A9   D0      08F9  2106          MOVL    IFB$L_ATJNLBUF(R9),R2       ; get address of journal record (RJR)
                     08FD  2107
5A  A2  22 A9   90       08FD  2108          MOVB    IFB$B_FAC(R9),RJR$B_FAC(R2) ; fill in specified file access
5B  A2  4E A9   90       0902  2109          MOVB    IFB$B_SHR(R9),RJR$B_SHR(R2) ; fill in specified file sharing
48  A2  70 A9   D0       0907  2110          MOVL    IFB$L_HBK(R9),RJR$L_ALLOC(R2) ; fill in high allocation
24  A2  08 A8   D0       090C  2111          MOVL    FAB$L_STS(R8),RJR$L_AT_STS(R2)  ; status
28  A2  0C A8   D0       0911  2112          MOVL    FAB$L_STV(R8),RJR$L_AT_STV(R2)  ; STV
2C  A2  18 A8   D0       0916  2113          MOVL    FAB$L_CTX(R8),RJR$L_AT_CTX(R2)  ; User definable CTX field
                     091B  2114
        04   BA       091B  2115 10$:     POPR    #^M<R2>                     ; restore work register
             05       091D  2116          RSB                                 ; to RM$MAPJNL
```

```
                                091E  2118          .SUBTITLE RMSAT_COM_RAB - Get common AT record data
                                091E  2119 ;++
                                091E  2120 ; AT_COM_RAB
                                091E  2121 ;
                                091E  2122 ;       This routine scarfs up and puts in the RMS journaling record the
                                091E  2123 ;       common RAB data at the beginning of an operation.
                                091E  2124 ;
                                091E  2125 ; Inputs:
                                091E  2126 ;
                                091E  2127 ;       R1      rjr operation id
                                091E  2128 ;       R8      RAB (the sucker is assumed to be probed.)
                                091E  2129 ;       R9      irab
                                091E  2130 ;       R10     ifab
                                091E  2131 ;
                                091E  2132 ; Outputs:
                                091E  2133 ;
                                091E  2134 ;       Some AT record RJR fields filled in.
                                091E  2135 ;
                                091E  2136 ;--
                                091E  2137
                                091E  2138 RMSAT_COM_RAB::
                                091E  2139
                          10    BB  091E  2140          PUSHR   #^M<R4>                         ; save work register
                    54    2C A9   D0  0920  2141          MOVL    IRB$L_ATJNLBUF(R9),R4           ; get MJB address
                          3C    13  0924  2142          BEQL    60$                             ; skip if none
                                    0926  2143
                    54    20 A4   DE  0926  2144          MOVAL   MJB$T_RJR(R4),R4                ; get RJR address in R4
                                    092A  2145
                3C  A4    04 A8   D0  092A  2146          MOVL    RAB$L_ROP(R8),RJR$L_AT_ROP(R4)  ; user's ROP
                40  A4    35 A8   90  092F  2147          MOVB    RAB$B_KRF(R8),RJR$B_AT_KRF(R4)  ; user's key of reference
                42  A4    1E A8   9C  0934  2148          MOVB    RAB$B_RAC(R8),RJR$B_AT_RAC(R4)  ; user's record access
                   05 A4      51   90  0939  2149          MOVB    R1,RJR$B_OPER(R4)               ; operation code
                2C  A4    18 A8   D0  093D  2150          MOVL    RAB$L_CTX(R8),RJR$L_AT_CTX(R4)  ; User context field
                                    0942  2151
                                    0942  2152 ;
                                    0942  2153 ; Probe key buffer before getting key.
                                    0942  2154 ;
                   01     1E A8   91  0942  2155          CMPB    RAB$B_RAC(R8),#RAB$C_KEY        ; keyed access?
                                1A  12  0946  2156          BNEQ    60$                             ; if not, no key size
                41  A4    34 A8   90  0948  2157          MOVB    RAB$B_KSZ(R8),RJR$B_AT_KSZ(R4)  ; user's key size
                                13  13  094D  2158          BEQL    60$                             ; if zero, no key
                                    094F  2159          IFNORD  RJR$B_AT_KSZ(R4),RAB$L_KBF(R8),60$ ; skip if can't get keybuffer
                                    0957  2160
                                    0957  2161 ;
                                    0957  2162 ; Copy search key into journal record
                                    0957  2163 ;
                                3E  BB  0957  2164          PUSHR   #^M<R1,R2,R3,R4,R5>             ; save MOVC3 registers
                          41    A4  28  0959  2165          MOVC3   RJR$B_AT_KSZ(R4),-              ; move KEY_SIZE number of chars
                          30    B8      095C  2166                  @RAB$L_KBF(R8),-                ;   from rab keybuffer
                          4C    A4      095E  2167                  RJR$T_AT_KEY(R4)                ;   to journal record
                                3E  BA  0960  2168          POPR    #^M<R1,R2,R3,R4,R5>             ; restore MOVC3 egisters
                                    0962  2169
                                10  BA  0962  2170 60$:     POPR    #^M<R4>                         ; restore work register
                                05  0964  2171 70$:     RSB                                     ; to caller
                                    0965  2172
                                    0965  2173          .END
```

```
$$.PSECT_EP                     = 00000000
$$RMSTEST                       = 0000001A
$$RMS_PBUGCHK                   = 00000010
$$RMS_TBUGCHK                   = 00000008
$$RMS_UMODE                     = 00000004
$$T1                            = 00000000
ACE$B_TYPE                      = 00000001
ACE$C_AIJNL                     = 00000003
ACE$C_ATJNL                     = 00000004
ACE$C_BIJNL                     = 00000002
ACE$C_JNLID                     = 00000008
ACE$M_HIDDEN                    = 00000400
ACE$M_NOPROPAGATE               = 00000800
ACE$M_PROTECTED                 = 00000200
ACE$T_RMSJNLNAM                 = 00000004
ACE$W_FLAGS                     = 00000002
ASS_DONE                          00000164 R      01
ATR$C_ADDACLENT                 = 0000001F
ATR$C_FNDACLTYP                 = 00000023
ATR$C_JOURNAL                   = 0000001D
ATR$C_UIC_RO                    = 0000001A
BDB$B_FLGS                      = 0000000A
BDB$L_ADDR                      = 00000018
BDB$L_IOSB                      = 00000048
BDB$T_JNLSEQ                    = 00000038
BDB$V_IOP                       = 00000002
BDB$W_NUMB                      = 00000014
CJF$ASSJNL                        ********  GX    01
CJF$DEASJNL                       ********  GX    01
CJF$FORCEJNL                      ********  GX    01
CJF$GETJNL                        ********  GX    01
CJF$WRITEJNL                      ********  GX    01
CJF$_AI                         = 00000003
CJF$_AT                         = 00000004
CJF$_BI                         = 00000002
CJF$_NONAME                       ********    X   01
CJF$_RU                         = 00000001
COMMON_FILE_AT                    000008F7 R      01
CTL$GL_PCB                        ********    X   01
CTL$GL_RUF                        ********    X   01
ERRJNS                            0000015A R      01
ERRMBC                            00000382 R      01
FAB$C_IDX                       = 00000020
FAB$C_REL                       = 00000010
FAB$C_SEQ                       = 00000000
FAB$L_CTX                       = 00000018
FAB$L_FOP                       = 00000004
FAB$L_STS                       = 00000008
FAB$L_STV                       = 0000000C
FAB$V_UFO                       = 00000011
FACILITY                          00000000 R      01
FIB$W_FID                       = 00000004
FORCE_JNL                         00000584 R      01
FWA$L_UIC                       = 00000028
FWA$Q_AIJNL                     = 000008D0
FWA$Q_ATJNL                     = 000008D8
FWA$Q_BIJNL                     = 000008C8
FWA$Q_DEVICE                    = 000000E0
FWA$Q_ID_DATE                   = 00000934
FWA$S_AIACE                     = 00000014
FWA$S_ATACE                     = 00000014
FWA$S_BIACE                     = 00000014
FWA$S_BIJNLN                    = 00000010
FWA$S_IDACE                     = 00000020
FWA$S_JNLID                     = 0000001C
FWA$T_AIACE                     = 000008F4
FWA$T_ATACE                     = 00000908
FWA$T_BIACE                     = 000008E0
FWA$T_FIBBUF                    = 000001F4
FWA$T_FID                       = 0000092C
FWA$T_IDACE                     = 0000091C
FWA$T_JNLID                     = 00000920
FWA$W_PRO                       = 0000002C
GET_JNL                           000000A1 R      01
IFB$B_BID                       = 00000008
IFB$B_BKS                       = 0000005E
IFB$B_FAC                       = 00000022
IFB$B_JNLFLG                    = 000000A0
IFB$B_JNLFLG2                   = 000000A2
IFB$B_ORGCASE                   = 00000023
IFB$B_RECVRFLGS                 = 000000A1
IFB$B_SHR                       = 0000004E
IFB$C_BID                       = 0000000B
IFB$C_IDX                       = 00000002
IFB$C_REL                       = 00000001
IFB$C_SEQ                       = 00000000
IFB$L_ATJNLBUF                  = 0000002C
IFB$L_EXTJNLBUF                 = 00000034
IFB$L_FWA_PTR                   = 00000038
IFB$L_HBK                       = 00000070
IFB$L_JNLBDB                    = 00000030
IFB$L_RJB                       = 000000A4
IFB$M_AI                        = 00000008
IFB$M_BI                        = 00000004
IFB$M_ONLY_RU                   = 00000001
IFB$M_RU                        = 00000002
IFB$V_AI                        = 00000003
IFB$V_AI_RECVR                  = 00000001
IFB$V_AT                        = 00000004
IFB$V_BI                        = 00000002
IFB$V_BIO                       = 00000005
IFB$V_BRO                       = 00000006
IFB$V_DONE_ASS_JNL              = 00000004
IFB$V_JNL                       = 00000001
IFB$V_ONLY_RU                   = 00000000
IFB$V_RU                        = 00000001
IFB$V_RUP                       = 00000002
IFB$V_WRTACC                    = 00000030
IFB$W_LRL                       = 00000052
IFB$W_MRS                       = 00000060
IO$_FORCE                       = 00000037
IO$_WRITEVBLK                   = 00000030
IRB$B_BID                       = 00000008
IRB$C_BID                       = 0000000A
```

```
IRB$L_ATJNLBUF          =  0000002C
IRB$L_IFAB_LNK          =  00000000
IRB$L_IOS               =  0000000C
IRB$L_JNLBDB            =  00000030
JBDB                    =  00000020
JTYP                    =  0000001C
MAPJNL                     0000038E  R      01
MJB$B_BID               =  00C00008
MJB$B_JNL               =  0000000C
MJB$C_BID               =  00000018
MJB$C_BLN               =  00000020
MJB$L_POINTER           =  00000014
MJB$Q_DESC              =  00000010
MJB$Q_IOSB              =  00000018
MJB$T_RJR               =  00000020
MJB$V_FILE              =  00000002
MJB$V_FORCE             =  C0000001
MJB$V_INIT              =  00000000
MJB$V_SYNCH_SHARE       =  00000003
MJB$W_FLAGS             =  0000000A
MODE                       00000002  R      01
OPEN_JNL                   00000269  R      01
PCB$C_STS               =  00000024
PCB$L_UIC               =  000000BC
PCB$V_RECOVER           =  0000001A
PSL$C_EXEC              =  00000001
RAB$B_KRF               =  00000035
RAB$B_KSZ               =  00000034
RAB$B_MBC               =  00000037
RAB$B_RAC               =  0000001E
RAB$C_BID               =  00000001
RAB$C_BLN               =  00000044
RAB$C_KEY               =  00000001
RAB$L_CTX               =  00C00018
RAB$L_KBF               =  00000030
RAB$L_RFAO              =  00000010
RAB$L_ROP               =  00000004
RAB$L_STV               =  0000000C
RAB$V_BIO               =  0000000B
RAB$W_RFA4              =  00000014
RBDB                    =  00000008
RJB$B_BID               =  00000008
RJB$C_BID               =  00000016
RJB$C_BLN               =  0000000C
RJB$Q_CHAN              =  00000000
RJB$V_AI                =  00000002
RJB$V_AT                =  00000003
RJB$V_BI                =  00000001
RJB$V_OPEN              =  00000004
RJB$V_RU                =  00000000
RJB$W_AICHAN            =  00000004
RJB$W_ATCHAN            =  00000006
RJB$W_BICHAN            =  00000002
RJB$W_FLAGS             =  0000000A
RJB$W_RUCHAN            =  00000000
RJR$B_AT_KRF            =  00000040
RJR$B_AT_KSZ            =  00000041
RJR$B_AT_RAC            =  00000042
RJR$B_ENTRY_TYPE        =  00000003
RJR$B_FAC               =  0000005A
RJR$B_FNS               =  00000058
RJR$B_OPER              =  00000005
RJR$B_ORG               =  00000004
RJR$B_SHR               =  0000005B
RJR$B_VERSION           =  00000002
RJR$C_AT_RECLEN         =  0000004C
RJR$C_AT_RECORD         =  00000006
RJR$C_BLKLEN            =  00000044
RJR$C_EXTLEN            =  0000007A
RJR$C_FILNAMLEN         =  000001C4
RJR$C_HDRLEN            =  00000038
RJR$C_IDX               =  00000002
RJR$C_MAPPING           =  00000001
RJR$C_MAXVER            =  00000002
RJR$C_RECLEN            =  00000048
RJR$C_REL               =  00000001
RJR$C_SEQ               =  00000000
RJR$L_ALLOC             =  00000048
RJR$L_AT_CTX            =  0000002C
RJR$L_AT_RFAO           =  00000044
RJR$L_AT_ROP            =  0000003C
RJR$L_AT_STS            =  00000024
RJR$L_AT_STV            =  00000028
RJR$S_FILENAME          =  00000100
RJR$T_AT_KEY            =  0000004C
RJR$T_FILENAME          =  000000C4
RJR$T_JNLID             =  00000008
RJR$W_AT_RFA4           =  00000048
RJR$_CLOSE              =  00000002
RJR$_OPEN               =  00000011
RM$ADJNLBUF                ********    X    01
RM$ALLOC_MJB               000006AA  RG     01
RM$ALLOC_RJB_BDB           000007C5  RG     01
RM$ASSJNL                  00000168  RG     01
RM$AT_COM_RAB              0000091E  RG     01
RM$AT_JNL_RECORD           00000857  RG     01
RM$CONJNL                  000002F0  RG     01
RM$DEAJNL                  000005F2  RG     01
RM$DSCJNL                  000005CD  RG     01
RM$FORCE_MJB               00000770  RG     01
RM$FRCJNL                  0000050A  RG     01
RM$GETBLK                  ********    X    01
RM$GETFILNAM               ********    X    01
RM$GETJNL                  00000004  RG     01
RM$MAPERR                  ********    X    01
RM$MAPJNL                  0000038C  RG     01
RM$MAPJNL_RU               00000388  RG     01
RM$RETBLK                  ********    X    01
RM$RETBLK1                 ********    X    01
RM$RETJNLBDB               ********    X    01
RM$RTVJNL                  000000F5  RG     01
RM$SETEFN                  ********    X    01
RM$STALL                   ********    X    01
RM$STALLAST                ********    X    01
```

```
RM$STALL_LOCK                   ********    X   01
RM$WRITE_MJB                    000006E3 RG     01
RM$WRTJNL                       0000044B RG     01
RM$WRTJNL_OBJ                   00000442 RG     01
RMS$_CJF                    =   0001C164
RMS$_FACILITY               =   00000001
RMS$_JNF                    =   0001C052
RMS$_JNS                    =   000187F4
RMS$_MBC                    =   00018734
RMS$_NOJ                    =   0001C154
RUCB$B_CTRL                 =   00000011
RUCB$V_ACTIVE               =   00000001
STS$S_FAC_NO                =   0000000C
STS$V_FAC_NO                =   00000010
SYS$GETTIM                      ********   GX   01
SYS$QIO                         ********   GX   01
UFO                             00000160 R      01
WRFLG$M_LOCK                =   00000010
WRFLG$M_OBJECT_ID           =   00000008
WRFLG$V_BI                  =   00000001
WRFLG$V_RUALSO              =   00000002
WRMOD$M_FORCE               =   00000040
WRTJNL                          00000452 R      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
RM$RMS_JOURNAL	00000965 (2405.)	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	30	00:00:00.05	00:00:00.93
Command processing	119	00:00:00.68	00:00:04.60
Pass 1	721	00:00:34.00	00:01:34.62
Symbol table sort	0	00:00:04.93	00:00:08.20
Pass 2	367	00:00:07.75	00:00:18.28
Symbol table output	29	00:00:00.26	00:00:00.73
Psect synopsis output	2	00:00:00.04	00:00:00.11
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	1270	00:00:47.71	00:02:07.48

The working set limit was 2400 pages.
189336 bytes (370 pages) of virtual memory were used to buffer the intermediate code.
There were 170 pages of symbol table space allocated to hold 3235 non-local and 104 local symbols.
2173 source lines were read in Pass 1, producing 20 object records in Pass 2.
51 pages of virtual memory were used to define 50 macros.

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

Macro library name                                  Macros defined
------------------                                  --------------
_$255$DUA28:[RMS.OBJ]RMS.MLB;1                              16
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                               4
_$255$DUA28:[SYSLIB]STARLET.MLB;2                           26
TOTALS (all libraries)                                      46

3505 GETS were required to define 46 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:RMOJOURNL/OBJ=OBJ$:RMOJOURNL MSRC$:RMOJOURNL/UPDATE=(ENH$:RMOJOURNL)+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

RM0RECLCK
LIS

RM0SET
LIS

RM0SCAN
LIS

RM0PRFLNM
LIS

RM0RCLCK2
LIS

RM0RABCHK
LIS

RM0RELEAS
LIS

RM0NAMSTR
LIS