```
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSSSSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSSSSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSSSSS
RRR         RRR  MMMMMM   MMMMMM  SSS
RRR         RRR  MMMMMM   MMMMMM  SSS
RRR         RRR  MMMMMM   MMMMMM  SSS
RRR         RRR  MMM   MMM   MMM  SSS
RRR         RRR  MMM   MMM   MMM  SSS
RRR         RRR  MMM   MMM   MMM  SSS
RRRRRRRRRRRR     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  SSSSSSSSSSSS
RRR         RRR  MMM         MMM  SSSSSSSSSSSS
RRR         RRR  MMM         MMM  SSSSSSSSSSSS
```

```
**FILE**ID**RMORECLCK
```

```
RRRRRRRR    MM      MM    000000    RRRRRRRR    EEEEEEEEEE    CCCCCCCC  LL            CCCCCCCC  KK      KK
RRRRRRRR    MM      MM    000000    RRRRRRRR    EEEEEEEEEE    CCCCCCCC  LL            CCCCCCCC  KK      KK
RR      RR  MMMM  MMMM  00      00  RR      RR  EE          CC          LL          CC          KK      KK
RR      RR  MMMM  MMMM  00      00  RR      RR  EE          CC          LL          CC          KK      KK
RR      RR  MM  MM  MM  00    0000  RR      RR  EE          CC          LL          CC          KK    KK
RR      RR  MM  MM  MM  00    0000  RR      RR  EE          CC          LL          CC          KK    KK
RRRRRRRR    MM      MM  00  00  00  RRRRRRRR    EEEEEEEE    CC          LL          CC          KKKKKK
RRRRRRRR    MM      MM  00  00  00  RRRRRRRR    EEEEEEEE    CC          LL          CC          KKKKKK
RR  RR      MM      MM  0000    00  RR  RR      EE          CC          LL          CC          KK    KK
RR  RR      MM      MM  0000    00  RR  RR      EE          CC          LL          CC          KK    KK
RR    RR    MM      MM  00      00  RR    RR    EE          CC          LL          CC          KK      KK      ....
RR    RR    MM      MM  00      00  RR    RR    EE          CC          LL          CC          KK      KK      ....
RR      RR  MM      MM    000000    RR      RR  EEEEEEEEEE    CCCCCCCC  LLLLLLLLLL    CCCCCCCC  KK      KK      ....
RR      RR  MM      MM    000000    RR      RR  EEEEEEEEEE    CCCCCCCC  LLLLLLLLLL    CCCCCCCC  KK      KK      ....


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            $BEGIN  RMORECLCK,000,RMSRMS0,<RECORD LOCK LIST (RLB) PROCESSING>
0000
0000    ;
0000    ;*************************************************************************
0000    ;*                                                                       *
0000    ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                              *
0000    ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.               *
0000    ;*  ALL RIGHTS RESERVED.                                                 *
0000    ;*                                                                       *
0000    ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED *
0000    ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE *
0000    ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER *
0000    ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY *
0000    ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY *
0000    ;*  TRANSFERRED.                                                         *
0000    ;*                                                                       *
0000    ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE *
0000    ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT *
0000    ;*  CORPORATION.                                                         *
0000    ;*                                                                       *
0000    ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  REL'ABILITY OF ITS *
0000    ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL               *
0000    ;*                                                                       *
0000    ;*                                                                       *
0000    ;*************************************************************************
0000    ;
0000    ;++
0000    ; Facility: rms32
0000    ;
0000    ; Abstract:
0000    ;               This module performs all the functions needed to implement
0000    ;               and process the record lock list (rlb).
0000    ;
0000    ; Environment:
0000    ;               Star processor running starlet exec.
0000    ;
0000    ; Author: E. H. Marison,        creation date:  28-SEP-1977
0000    ;
0000    ; Modified By:
0000    ;
0000    ;       V03-014 JEJ0043         J E Johnson             21-Jun-1984
0000    ;               Tweak the instruction stream for a little
0000    ;               performance boost.
0000    ;
0000    ;       V03-013 JWT0160         Jim Teague              29-Feb-1984
0000    ;               Remove calls to RM$DEALLEFN.
0000    ;
0000    ;       V03-012 JWT0141         Jim Teague              11-Nov-1983
0000    ;               Change IFB$V_RUM to IFB$V_ONLY_RU
0000    ;
0000    ;       V03-011 DAS0004         David Solomon           27-Jun-1983
0000    ;               Correct typo in V03-010.
0000    ;
0000    ;       V03-010 KPL0008         Peter Lieberwirth       21-Jun-1983
0000    ;               Set LCK$M_PROTECT if file can be recovery unit journaled.
0000    ;               Set LCK$M_RECOVER if recovery unit is being recovered.  These
0000    ;               flags are used to coordinate failover between the lock manager
```

```
0000  ;                  and the RCP.
0000  ;
0000  ;          V03-009 KPL0007         Peter Lieberwirth       20-Jun-1983
0000  ;                  Change some references to JNLFLG to JNLFLG2.
0000  ;
0000  ;          V03-008 JWH0198         Jeffrey W. Horn         21-Mar-1983
0000  ;                  Restructure RULOCK list so that the RLBs hang off
0000  ;                  of FLBs (File lock blocks).
0000  ;                  Add RM$SAVE_FL to save the file lock on the RULOCK
0000  ;                  list in a FLB.
0000  ;
0000  ;          V03-007 DAS0003         David Solomon           24-Feb-1983
0000  ;                  Add timeout on record lock wait capability.
0000  ;
0000  ;          V03-006 JWH0181         Jeffrey W. Horn         03-Feb-1983
0000  ;                  Add check for IFB$V_RU_RLK to QUERY.
0000  ;
0000  ;          V03-005 JWH0175         Jeffrey W. Horn         26-Jan-1983
0000  ;                  Fix bad branch destination in RUSCAN when the RLB
0000  ;                  is not found.
0000  ;                  Always make sure IFB$V_NO_Q_WAIT is clear when exiting
0000  ;                  RM$QUERY_PROC and RM$QUERY_LCK.
0000  ;
0000  ;          V03-004 JWH0169         Jeffrey W. Horn         10-Jan-1983
0000  ;                  If giving back a Recovery Unit held lock then return
0000  ;                  alternate status of OK_RULK.
0000  ;
0000  ;                  Add RM$QUERY_PROC to search for a lock on the RU list regardless
0000  ;                  of stream and then if not found join QUERY.
0000  ;
0000  ;                  For QUERY, if we had to do an $ENQ, then always deque the
0000  ;                  lock, regardless of RU.
0000  ;
0000  ;                  If the bit IFB$V_RU_RLK is set then do not perform $ENQs.
0000  ;
0000  ;                  If the bit IRB$V_NO_Q_WAIT is set then do not wait on $ENQs
0000  ;                  within QUERY.
0000  ;
0000  ;          V03-003 JWH0001         Jeffrey W. Horn         19-Aug-1982
0000  ;                  Put in Recovery Unit Lock support:
0000  ;
0000  ;                          1.      If stream releases lock in RU hold lock
0000  ;                                  in PIO$GL_RULOCK list.
0000  ;
0000  ;                          2.      If same steam trys to re-access lock
0000  ;                                  released in that RU, give it back after
0000  ;                                  conversion if necessary.
0000  ;
0000  ;                          3.      Put in RM$RU_UNLOCK routine to release
0000  ;                                  all locks held for the durration of an RU.
0000  ;
0000  ;                          4.      RLB$L_OWNER now contains the value of
0000  ;                                  IRB$L_IDENT for the owning stream, which
0000  ;                                  is a unique (for the life of the process)
0000  ;                                  identifier for each stream.
0000  ;
0000  ;          V03-002 KBT0307         Keith B. Thompson       25-Aug-1982
```

```
0000 ;                  Reorganize psects
0000 ;
0000 ;          V03-001 CDS0001         C Saether               1-Mar-1982
0000 ;                  Save R2 when stalling for a record lock.
0000 ;
0000 ;          V02-011 KPL0006         Peter Lieberwirth       21-Oct-1981
0000 ;                  Add additional entry points so that query_lck and unlock
0000 ;                  will return a RNL status in those places where a REA lock
0000 ;                  is held and the caller expects to get away with doing an
0000 ;                  update or delete after a get or find (that only applied a
0000 ;                  REA lock).  This is important because several streams can
0000 ;                  hold a REA lock on a single record, so if any can update
0000 ;                  the record, consistency is lost.
0000 ;
0000 ;                  This wasn't a problem before because REA locks weren't
0000 ;                  really being applied properly (see next paragraph), and REA
0000 ;                  will now be permitted in files opened for write access.
0000 ;
0000 ;                  Fix implementation of REA lock; it should map to PR (protected
0000 ;                  read) not PW (protected write).  If both REA and RLK are set,
0000 ;                  REA takes precedence.
0000 ;
0000 ;                  Remove bugcheck on DEQ_S failure, it doesn't do anything good.
0000 ;
0000 ;                  Fix some commentary.
0000 ;
0000 ;          V02-010 kpl0005         Peter Lieberwirth       30-Sep-1981
0000 ;                  Always release curbdb on record lock stall.
0000 ;
0000 ;          V02-009 kpl0004         Peter Lieberwirth       3-Aug-1981
0000 ;                  Make the following changes:
0000 ;
0000 ;                  1  move RLBs to a list off the IRAB from the IFAB
0000 ;                  2. zero CURBDB so GET doesn't try to deaccess again
0000 ;                     on errors (WAT only)
0000 ;                  3. when WAT not taken, deallocate the sync efn as well
0000 ;                     as set it
0000 ;                  4. redesign and fix bugs regarding RRL and REA
0000 ;                  5. remove RM$UNLOCK_ALT entry point since its no longer
0000 ;                     needed
0000 ;
0000 ;          V02-008 mcn0006         Maria del C. Nasr       23-Jul-1981
0000 ;                  record id size changes from a byte to a word
0000 ;
0000 ;          V02-007 kpl0003         Peter Lieberwirth       7-Jul-1981
0000 ;                  Add testpoints to count number of times RM$LOCK and
0000 ;                  RM$QUERY_LCK are called.  Also add a testpoint to see
0000 ;                  how many times we do a wait on a record lock conflict.
0000 ;                  (This last depends on user setting the ROP WAT bit.)
0000 ;
0000 ;          V02-006 kpl0002         Peter Lieberwirth       5-Jan-1981
0000 ;                  Rewrite to use $enq/$deq to lock and unlock records.
0000 ;                  rm$query_lock can now return ok_rrl if ROP function
0000 ;                  RRL is specified and record is locked against readers.
0000 ;
0000 ;          V02-005 REFORMAT        Maria del C. Nasr       24-Jul-1980
0000 ;
```

```
0000   ;       V004    RAN0003         R A NEWELL              9-nov-1978      10:14
0000   ;               file sharing code enhancements
0000   ;
0000   ; Revision History:
0000   ;
0000   ;       L F LAVERDURE,      9-oct-1978  17:16
0000   ;               add shared file code
0000   ;
0000   ;--
0000   ;
```

```
          0000          .SBTTL  DECLARATIONS
          0000
          0000  ;
          0000  ; Include Files:
          0000  ;
          0000
          0000          $RABDEF
          0000          $RLBDEF
          0000          $IRBDEF
          0000          $IFBDEF
          0000          $RMSDEF
          0000          $SFSBDEF
          0000          $SSDEF
          0000          $ENQDEF
          0000          $LCKDEF
          0000          $FLBDEF
          0000
          0000  ;
          0000  ; Macros:
          0000  ;
          0000  ;
          0000  ; Equated Symbols:
          0000  ;
          0000
00000020  0000          ROP             =       RAB$L_ROP*8
          0000
          0000
          0000  ;
          0000  ; Own Storage:
          0000  ;
```

```
0000           .SBTTL  RM$LOCK AND RM$QUERY_LCK
0000
0000    ;++
0000    ;
0000    ; RM$LOCK       - make entry in the lock list for specified record
0000    ; RM$QUERY_LCK  - search rlb for specified record and report status
0000    ;
0000    ; Calling sequence:
0000    ;               bsbw    rm$lock
0000    ;               bsbw    rm$query_lck
0000    ;
0000    ;
0000    ; Input Parameters:
0000    ;
0000    ;       r11     impure area address
0000    ;       r10     ifab address
0000    ;       r9      irab address *** please note always irab ***
0000    ;       r8      rab address
0000    ;       r1      1'st and 2'nd word of record's rfa
0000    ;       r2      3'rd word of record's rfa
0000    ;                seq f.o.        offset (always positive value)
0000    ;                relative f.o.   always 0
0000    ;                index f.o.      low byte = record id
0000    ;
0000    ;
0000    ; Implied Input:
0000    ;       rm$lock
0000    ;               the wat bit in rop (ie queue the request if it can't
0000    ;                                   be granted immediately)
0000    ;               the tmo bit in rop (if WAT is set, wait for a specific amount
0000    ;                                   of time before returning timeout error).
0000    ;               the rlk bit in rop (ie lock for write, allow readers)
0000    ;               the rea bit in rop (ie lock for read, allow readers)
0000    ;       rm$query_lck
0000    ;               the rrl bit in rop (ie read record regardless of lock)
0000    ;
0000    ; Output Parameters:
0000    ;
0000    ;       r3 is destroyed
0000    ;
0000    ;       r0      internal rms status code
0000    ;               rm$lock:
0000    ;                rms$_suc&^xffff        record lock entry made
0000    ;                rms$_ok_wat&^xffff     record lock entry was made, but we had
0000    ;                                       to wait to get it, caller must reaccess
0000    ;                                       buffer
0000    ;                rms$_ok_alk&^xffff     record was already locked by caller
0000    ;                rms$_rlk&^xffff        record is locked by another
0000    ;                                       process-stream
0000    ;                rms$_dme&^xffff        could not get space for new rlb block
0000    ;                rms$_tmo&^xffff        record lock timed out
0000    ;
0000    ;               rm$query_lck:
0000    ;                rms$_suc&^xffff        record not locked
0000    ;                rms$_ok_alk&^xffff     record was already locked by caller
0000    ;                rms$_ok_rlk&^xffff     record is locked by another
0000    ;                                       process-stream but read is allowed
```

```
                        0000      ;                 rms$_ok_rrl&^xffff     record is locked by another
                        0000      ;                                        process-stream, but RRL overrides lock
                        0000      ;                 rms$_rlk&^xffff        record is locked by another
                        0000      ;                                        process-stream
                        0000      ;
                        0000      ;
                        0000      ; Side Effects:
                        0000      ;
                        0000      ;       If success code rms$_ok_wat is returned from RMSLOCK, record was
                        0000      ;       successfully locked, but access to the buffer was given up to do
                        0000      ;       a stall.  The caller must reaccess the buffer.
                        0000      ;
                        0000      ;--
                        0000
                        0000      RMSLOCK::
                        0000              $TSTPT  LOCK                          ; count this call
                        0006                                                    ;
 0A 00A2 CA   02   E1   0006              BBC     #IFB$V_RUP,IFB$B_JNLFLG2(R10),5$ ; branch if not in RU
       028E   30        000C              BSBW    RUSCAN                        ; scan RU lock list
      5D 50   E8        000F              BLBS    R0,50$                        ; get out if one found
         50   D5        0012              TSTL    R0                            ; was it an error instead?
         59   12        0014              BNEQ    50$                           ; yes, get out.
       0239   30        0016      5$:     BSBW    SCAN                          ; scan lock list
      0C 50   E9        0019              BLBC    R0,10$                        ; only success codes are ok_alk and ok_rlk
   50  8039 8F  B1      001C              CMPW    #<RMS$_OK_ALK&^XFFFF>,R0      ; already locked?
         4C   13        0021              BEQL    50$                           ; branch if yes
                        0023              RMSERR  RLK                           ; otherwise change it to rlk
         53   D4        0028      10$:    CLRL    R3                            ; if we get here don't want this rlb
   50  81A0 8F  B1      002A              CMPW    #<RMS$_RNL&^XFFFF>,R0         ; record not in local lock list?
         3E   12        002F              BNEQ    50$                           ; exit if record is locked
       034F   30        0031              BSBW    GET_RLB                       ; find an RLB
      38 50   E9        0034              BLBC    R0,50$                        ; branch on any error
                        0037                                                    ;
   05 68   2C   E1      0037              BBC     #RAB$V_LV2+ROP,(R8),15$       ; propigate LV2 bit to RLB
                        003B              SSB     #RLB$V_LV2,RLB$B_FLAGS(R3)
                        0040
   05 68   22   E1      0040      15$:    BBC     #RAB$V_REA+ROP,(R8),20$       ; is it a REA type lock?  branch if no
           03   E3      0044              BBCS    #RLB$V_PR,-                   ; map REA to protected read
   09 0B A3             0046                      RLB$B_FLAGS(R3),30$           ;
                        0049                                                    ;
   05 68   33   E1      0049      20$:    BBC     #RAB$V_RLK+ROP,(R8),30$       ; is it a RLK type lock?  branch if no
           02   E3      004D              BBCS    #RLB$V_PW,-                   ; map RLK to protected write
   00 0B A3             004F                      RLB$B_FLAGS(R3),30$           ;
                        0052
                        0052      ;+
                        0052      ; Save record lock wait/timeout information in RLB. If the ROP WAT
                        0052      ; bit is not set, don't even look at the TMO bit.
                        0052      ;-
                        0052
   13 68   31   E1      0052      30$:    BBC     #RAB$V_WAT+ROP,(R8),40$       ; Should we wait on record lock?
           00   E3      0056              BBCS    #RLB$V_WAIT,-                 ; Yes, propagate bit to RLB.
   00 0B A3             0058                      RLB$B_FLAGS(R3),35$           ;
                        005B
   0A 68   39   E1      005B      35$:    BBC     #RAB$V_TMO+ROP,(R8),40$       ; Is a timeout specified?
           07   E3      005F              BBCS    #RLB$V_TMO,-                  ; Propagate bit to RLB.
   00 0B A3             0061                      RLB$B_FLAGS(R3),37$
      1F A8   90        0064      37$:    MOVB    RAB$B_TMO(R8),-               ; Store timeout value in RLB.
```

```
    0A A3      0067   328                   RLB$B_TMO(R3)
               0069   329
     00BA  30  0069   330 40$:    BSBW      DO_ENQ                          ; lock the record
     00A6  31  006C   331         BRW       RRL                             ; go check for read-regardless
           05  006F   332 50$:    RSB                                       ; return all status to caller
```

```
                              0070  ;++
                              0070  ; RM$QUERY_LCK
                              0070  ;
                              0070  ;       If the record is not locked locally, see if its locked by
                              0070  ;       another process by requesting a lock on it.  If the lock
                              0070  ;       is granted, the record may be read.  Also, immediately unlock
                              0070  ;       the record if lock granted, so extraneous junk doesn't fill
                              0070  ;       up the lock database.
                              0070  ;
                              0070  ; RM$QUERY_HARD
                              0070  ;
                              0070  ;       Same as QUERY_LCK, but map OK_ALK when lock is REA type to RNL so
                              0070  ;       any writers of the file holding a REA lock on the record can't get
                              0070  ;       away with updating or deleting it.
                              0070  ;
                              0070  ;       Algorithmn for query_lock
                              0070  ;
                              0070  ;               first try PR - if this succeeds, it means there
                              0070  ;               was no lock, and its OK to read.
                              0070  ;
                              0070  ;               if PR fails, it means either an EX or PW lock is
                              0070  ;               held on the record, so try CR, with WAIT if the
                              0070  ;               user said to.  If CR succeeds, then the lock must
                              0070  ;               have been PW, so its OK to read.
                              0070  ;
                              0070  ;       Also, read-regardless of lock (RRL) is handled here.  If all
                              0070  ;       indications are that the record is locked, then if RRL is
                              0070  ;       specified, access to that record is permitted.
                              0070  ;
                              0070  ;
                              0070  ; RM$QUERY_PROC
                              0070  ;
                              0070  ;       Scan RU Lock list for lock regardless of stream, if found
                              0070  ;       return OK_RULK otherwise join RM$QUERY_LCK code.
                              0070  ;
                              0070  ;--
                              0070
                              0070 RM$QUERY_HARD::
                              0070          $TSTPT  QUERY_LCK                  ; count this call
              01D9    30      0076          BSBW    SCAN                       ; scan for record
    50   8039 8F      B1      0079          CMPW    #<RMS$_OK_ALK&^XFFFF>,R0; is record locked by caller?
                0A    12      007E          BNEQ    10$                        ; if NEQ no
                03    E1      0080          BBC     #RLB$V_PR,-                ; yes, but is it only REA?
          05 0B A3            0082                  RLB$B_FLAGS(R3),10$        ;  branch if not REA
                              0085          RMSERR  RNL                        ; map OK_ALK to RNL if its locked REA
                      05      008A 10$:     RSB                                ; return to caller
                              008B
                              008B RM$QUERY_PROC::
                              008B          $TSTPT  QUERY_LCK                  ; count this call
              02B8    30      0091          BSBW    PRSCAN                     ; scan RU list for lock
             0A 50    E9      0094          BLBC    R0,RM$QUERY_LCK            ; continue with Query lock if not there
                              0097          RMSSUC  OK_RULK                    ; set alternate success
                              009C          CSB     #IRB$V_NO_Q_WAIT,(R9)      ; make sure this bit is clear
                      05      00A0          RSB
                              00A1
                              00A1 RM$QUERY_LCK::
                              00A1          $TSTPT  QUERY_LCK                  ; count this call
          01A8   30  00A7               BSBW    SCAN                        ; scan for record
  50   81A0 8F   B1  00AA               CMPW    #<RMS$_RNL&^XFFFF>,R0       ; if RNL, record may be locked by
                     00AF                                                   ; another process
            20   12  00AF               BNEQ    10$                         ; return status of scan
            03   E0  00B1               BBS     #IFB$V_RU_RLK,-             ; get out if 'fake' record locking
      1A 00A2 CA     00B3                       IFB$B_JNLFLG2(R10),10$
          02C9   30  00B7               BSBW    GET_RLB                     ; find an RLB to use
         14 50   E9  00BA               BLBC    R0,10$                      ; pass along possible DME error
            08   90  00BD               MOVB    #RLB$M_PR,-                 ;
         0B A3       00BF                       RLB$B_FLAGS(R3)             ; ask only to read
            51   DD  00C1               PUSHL   R1                          ; save RFA across enq
            61   10  00C3               BSBB    DO_ENQ                      ; go try to lock the record
          51 8ED0    00C5               POPL    R1                          ; restore RFA
         0B 50   E9  00C8               BLBC    R0,20$                      ; if error, go try CR
          0326   30  00CB               BSBW    DEQUE_QUERY                 ; got the lock, so give it up now
                     00CE               RMSSUC                              ; permission to read record
                     00D1       10$:    CSB     #IRB$V_NO_Q_WAIT,(R9)       ; make sure this bit is clear
                 05  00D5               RSB                                 ; return to caller
                     00D6                                                   ;
          02E3   30  00D6       20$:    BSBW    SETOWNRFA                   ; reset ownership and rfa
                     00D9                                                   ; try again for lor
            02   90  00D9               MOVB    #RLB$M_CR,-                 ;
         0B A3       00DB                       RLB$B_FLAGS(R3)             ; try for concurrent read
                     00DD
                     00DD       ;+
                     00DD       ; Save record lock wait/timeout information in RLB. If the ROP WAT
                     00DD       ; bit is not set, don't even look at the TMO bit.
                     00DD       ;-
                     00DD
    17 69       38   E4  00DD   30$:    BBSC    #IRB$V_NO_Q_WAIT,(R9),40$ ; branch if no queuing
    13 68       31   E1  00E1           BBC     #RAB$V_WAT+ROP,(R8),40$ ; is queuing disabled
                         00E5           SSB     #RLB$V_WAIT,-
                         00E5                   RLB$B_FLAGS(R3)
                         00EA
    0A 68       39   E1  00EA           BBC     #RAB$V_TMO+ROP,(R8),40$ ; Is a timeout specified?
                         00EE           SSB     #RLB$V_TMO,-            ; Propagate bit to RLB.
                         00EE                   RLB$B_FLAGS(R3)
         1F A8   90  00F3               MOVB    RAB$B_TMO(R8),-             ; Store timeout value in RLB.
         0A A3       00F6                       RLB$B_TMO(R3)
                     00F8
            2C   10  00F8       40$:    BSBB    DO_ENQ                      ; go try to lock the record
         15 50   E9  00FA               BLBC    R0,50$                      ; branch on record lock error
            50   DD  00FD               PUSHL   R0                          ; save status of lock operation
          02F2   30  00FF               BSBW    DEQUE_QUERY                 ; go unlock record we just locked
          50 8ED0    0102               POPL    R0                          ; restore lock status
  50   8061 8F   B1  0105               CMPW    #<RMS$_OK_WAT&^XFFFF>,R0; did we wait for the record?
            05   13  010A               BEQL    45$                         ; if eql we waited for the lock
                     010C               RMSSUC  OK_RLK                      ; record locked, but ok to read
                 05  0111       45$:    RSB                                 ; return to caller
                     0112
          02BE   30  0112       50$:    BSBW    RESET_RLB                   ; free the RLB
                     0115
                     0115       ;
                     0115       ; read regardless of lock:
                     0115       ;
                     0115       ; If the record is locked, then if the user specified RRL
                     0115       ; we'll return the record with the code OK_RRL.
```

```
                           0115     ;
                           0115     RRL:                                            ;
   50    82AA 8F     B1    0115             CMPW    #<RMS$_RLK&^XFFFF>,R0          ; is the error RLK?
                 09  12    011A             BNEQ    10$                            ; if neq no, don't try read regardless
     05 68       23  E1    011C             BBC     #RAB$V_RRL+ROP,(R8),10$        ; has rrl been specified
                           0120             RMSSUC  OK_RRL                         ; yes
                     05    0125     10$:    RSB                                    ; return to caller
```

```
                       0126              .SBTTL  DO_ENQ
                       0126      ;++
                       0126      ; DO_ENQ
                       0126      ;       - build the enq on the stack, perform it, and handle any errors
                       0126      ;
                       0126      ; Calling Sequence:
                       0126      ;                       bsbb    do_enq
                       0126      ;
                       0126      ; Input Parameters
                       0126      ;
                       0126      ;       r10     ifab address
                       0126      ;       r9      irab address
                       0126      ;       r8      rab/fab address (not needed by this routine, but
                       0126      ;                               must preserve this register)
                       0126      ;       r3      rlb address
                       0126      ;
                       0126      ; Output Parameters:
                       0126      ;
                       0126      ;       r0      status of enq
                       0126      ;       rlb fields filled in: lock_id, lksb
                       0126      ;
                       0126      ;
                       0126      ; Side Effects:
                       0126      ;
                       0126      ;       If success, record is locked.  If status is OK_WAT, bucket has been
                       0126      ;       deaccessed, and caller must reaccess.
                       0126      ;
                       0126      ;--
                       0126
                       0126     DO_ENQ:                                         ; perform the sys$enq
                       0126              RMSSUC                                 ; assume success for the next check
  00A2 CA    03    E1  0129              BBC     #IFB$V_RU_RLK,IFB$B_JNLFLG2(R10),-
             06        012E                      5$                             ; only do enq if sharing file
                       012F              SSB     #RLB$V_FAKE,RLB$B_FLAGS(R3) ; set as 'fake' RLB
                   05  0134              RSB                                    ; return to caller
                       0135
             33    E1  0135     5$:      BBC     #IFB$V_NORECLK,-               ; only do enq if record locking
          01 6A        0137                      (R10),10$                      ;
                   05  0139              RSB                                    ; return to caller
                       013A
                       013A      ;
                       013A      ; Build RESNAM and descriptor for it on stack.  Warning, if any parameters to
                       013A      ; SYS$ENQ are added or removed prior to RESNAM descriptor address, offset to
                       013A      ; RESNAM descriptor will change...
                       013A      ;
                       013A
       5E    10    C2  013A     10$:     SUBL2   #16,SP                         ; make room for record RESAM and
                       013D                                                     ; descriptor on stack
       6E    08    D0  013D              MOVL    #8,(SP)                        ; length of RESNAM in descriptor
                       0140                                                     ; RFA is only 6, but $enq optimizes 8
  04 AE   08 AE    DE  0140              MOVAL   8(SP),4(SP)                    ; address of RESNAM in descriptor
  08 AE   06 A3    3C  0145              MOVZWL  RLB$W_RFA4(R3),8(SP)           ; second longword of RESNAM is 3rd
                       014A                                                     ; word of RFA
  0C AE   0C A3    D0  014A              MOVL    RLB$L_RFA0(R3),12(SP)          ; first longword of RESNAM is 1st
                       014F                                                     ; longword of RFA
                       014F      ;++
                       014F      ;
```

```
                        014F   513 ; Perform the sys$enq function, building the parameter list on the stack.
                        014F   514 ;
                        014F   515 ; First, verify assumptions about order of arguments on stack
                        014F   516 ;
                        014F   517 ;--
                        014F   518
                        014F   519          ASSUME ENQ$_EFN        EQ <ENQ$_LKMODE - 4>
                        014F   520          ASSUME ENQ$_LKMODE     EQ <ENQ$_LKSB   - 4>
                        014F   521          ASSUME ENQ$_LKSB       EQ <ENQ$_FLAGS  - 4>
                        014F   522          ASSUME ENQ$_FLAGS      EQ <ENQ$_RESNAM - 4>
                        014F   523          ASSUME ENQ$_RESNAM     EQ <ENQ$_PARID  - 4>
                        014F   524          ASSUME ENQ$_PARID      EQ <ENQ$_ASTADR - 4>
                        014F   525          ASSUME ENQ$_ASTADR     EQ <ENQ$_ASTPRM - 4>
                        014F   526          ASSUME ENQ$_ASTPRM     EQ <ENQ$_BLKAST - 4>
                        014F   527          ASSUME ENQ$_BLKAST     EQ <ENQ$_ACMODE - 4>
                        014F   528          ASSUME ENQ$_ACMODE     EQ <ENQ$_PROT   - 4>
                        014F   529
                        014F   530          ASSUME ENQ$_NARGS      EQ 11
                        014F   531
          7E      7C    014F   532          CLRQ    -(SP)                   ; let the protection and mode default
          7E      D4    0151   533          CLRL    -(SP)                   ; no blocking ast for records
          59      DD    0153   534          PUSHL   R9                      ; astprm = irab
     0000'CF      DF    0155   535          PUSHAL  W^RM$STALLAST           ; ast address
  50   78 AA      D0    0159   536          MOVL    IFB$L_SFSB_PTR(R10),R0  ; get SFSB address
          3A      13    015D   537          BEQL    27$                     ; error if there is none
       30 A0      DD    015F   538          PUSHL   SFSB$L_LOCK_ID(R0)      ; parent_id is SFSB lock id
       18 AE      DF    0162   539          PUSHAL  24(SP)                  ; resnam descriptor address
          1C      DD    0165   540          PUSHL   #LCK$M_SYNCSTS!LCK$M_NOQUEUE!LCK$M_SYSTEM
                        0167   541                                          ; don't take ast if enq completes fast
                        0167   542                                          ; don't wait unless user tells us to
                        0167   543                                          ; lock is not to be qualified by group
          03      93    0167   544          BITB    #<IFB$M_RU!IFB$M_ONLY_RU>,-      ; recovery unit journaled?
     00A0 CA            0169   545                  IFB$B_JNLFLG(R10)       ;
          04      13    016C   546          BEQL    17$                     ; if EQL not marked for RU journaling
                        016E   547          SSB     #LCK$V_PROTECT,(SP)     ; lock is protected for failover
                        0172   548
                        0172   549          ASSUME  IFB$V_RU_RECVR EQ 0
                        0172   550
  04 00A1 CA      E9    0172   551 17$:     BLBC    IFB$B_RECVRFLGS(R10),18$; RU recovery in progress on this file?
                        0177   552          SSB     #LCK$V_RECOVER,(SP)     ; lock is interesting during failover
                        017B   553
                        017B   554          ASSUME  RLB$V_WAIT EQ 0
                        017B   555
    03 0B A3      E9    017B   556 18$:     BLBC    RLB$B_FLAGS(R3),20$     ; branch if not ok to wait
     6E   04      CA    017F   557          BICL2   #LCK$M_NOQUEUE,(SP)     ; wait for lock if not immediately
                        0182   558                                          ; available
          04      E1    0182   559 20$:     BBC     #RLB$V_CONV,-
    03 0B A3            0184   560                  RLB$B_FLAGS(R3),25$     ; branch if not converting a lock
     6E   02      C8    0187   561          BISL2   #LCK$M_CONVERT,(SP)     ; set lock convert
       14 A3      DF    018A   562 25$:     PUSHAL  RLB$L_LKSB(R3)          ; address of lock status block
          05      DD    018D   563          PUSHL   #LCK$K_EXMODE           ; assume exclusive lock for now
          02      E1    018F   564          BBC     #RLB$V_PW,-             ;
    08 0B A3            0191   565                  RLB$B_FLAGS(R3),30$     ; is it protected write?
     6E   04      D0    0194   566          MOVL    #LCK$K_PWMODE,(SP)      ; make lkmode protected write
          15      11    0197   567          BRB     50$                     ; go allocate efn
        0345      31    0199   568 27$:     BRW     ERRENQ                  ; branch aid
          03      E1    019C   569 30$:     BBC     #RLB$V_PR,-             ;
```

```
             05 0B A3        019E   570                     RLB$B_FLAGS(R3),40$      ; is it protected read?
                6E  03   D0  01A1   571             MOVL    #LCK$K_PRMODE,(SP)       ; make lkmode protected read
                    08   11  01A4   572             BRB     50$                      ; go allocate efn
                    01   E1  01A6   573 40$:        BBC     #RLB$V_CR,-              ;
             03 0B A3        01A8   574                     RLB$B_FLAGS(R3),50$      ; is it concurrent read?
                6E  01   D0  01AB   575             MOVL    #LCK$K_CRMODE,(SP)       ; make lkmode concurrent read
                 FE4F'   30  01AE   576 50$:        BSBW    RM$SETEFN                ; allocate a synchronous event flag
                             01B1   577                                              ;
       00000000'9F  0B   FB  01B1   578             CALLS   #11,@#SYS$ENQ            ; do the enq
                5E  10   C0  01B8   579             ADDL2   #16,SP                   ; pop RESNAM and its descriptor
                 75 50   E9  01BB   580             BLBC    R0,110$                  ; branch on error
           0689 8F  50   B1  01BE   581             CMPW    R0,#SS$_SYNCH            ; synchronous completion?
                    11   12  01C3   582             BNEQ    90$                      ; no, go stall
                             01C5   583                                              ;
                             01C5   584             $SETEF_S        IRB$B_EFN(R9)    ; set event flag we didn't stall for
                             01CF   585             RMSSUC                           ; indicate successful lock
                         05  01D2   586             RSB                              ; and return
                             01D3   587                                              ;
                  030B   31  01D3   588 80$:        BRW     ERRENQ                   ; branch aid
                             01D6   589
                             01D6   590 ;+
                             01D6   591 ; If timeout on record lock specified, set up timer.
                             01D6   592 ;-
                             01D6   593
                    07   E1  01D6   594 90$:        BBC     #RLB$V_TMO,-             ; If timeout not specified, skip this.
             20 0B A3        01D8   595                     RLB$B_FLAGS(R3),95$
                 FE22'   30  01DB   596             BSBW    RM$SET_LOCK_TMO          ; Set timer for lock.
                 1A 50   E8  01DE   597             BLBS    R0, 95$                  ; If successful, continue.
                    50   DD  01E1   598             PUSHL   R0                       ; Save $SETIMR error status
                             01E3   599             $DEQ_S  LKID=RLB$L_LOCK_ID(R3)   ; Else $SETIMR failed; cancel $ENQ.
                 50 8ED0     01F1   600             POPL    R0                       ; Restore error status
                             01F4   601             RMSERR  TMR, R1                  ; Unexpected $SETIMR error.
                    49   11  01F9   602             BRB     125$                     ; Go map error and exit.
                             01FB   603
                    1C   BB  01FB   604 95$:        PUSHR   #^M<R2,R3,R4>            ; save registers
                             01FD   605
                             01FD   606 ;++
                             01FD   607 ; Release curbdb because we don't want the bucket locked while we are waiting
                             01FD   608 ; (possibly for a long time) for the record.  No one can even get in to unlock
                             01FD   609 ; the record if we have the bucket locked.
                             01FD   610 ;
                             01FD   611 ; The extremely important assumption is made here that no STALL will be done in
                             01FD   612 ; RM$RELEASF.  No bucket will be written for example.  This call should only
                             01FD   613 ; deaccess the buffer.  If this assumption is invalid, then all $ENQ
                             01FD   614 ; synchronization is blown because there aren't enough EFNs to go around.
                             01FD   615 ;--
                             01FD   616
          54    20 A9    D0  01FD   617             MOVL    IRB$L_CURBDB(R9),R4      ; point to current bdb
                    48   13  0201   618             BEQL    NOBDB                    ; error if there is none
                             0203   619                                              ;
                 20 A9   D4  0203   620             CLRL    IRB$L_CURBDB(R9)         ; zero CURBDB so error pats don't try
                             0206   621                                              ; to release it again
                             0206   622             $TSTPT  REC_WAT                  ; count number of times WAT waits
                             020C   623                                              ;
                    53   D4  020C   624             CLRL    R3                       ; no flags for rm$release
                 FDEF'   30  020E   625             BSBW    RM$RELEASE               ; deaccess the buffer - no IO
                 FDEC'   30  0211   626 100$:       BSBW    RM$STALL                 ; await completion of enqueue
```

```
            1C    BA  0214               POPR    #^M<R2,R3,R4>           ; restore registers
                      0216
                      0216       ;+
                      0216       ; If a timer was still outstanding, cancel the request.
                      0216       ;
                      0216       ; Note: if the timer fires after the BBCC instruction, but before the $CANTIM,
                      0216       ; the timer AST routine will simply exit since the RLB$V_TIMER_INPROG flag will
                      0216       ; be clear. If the timer fires before the BBCC instruction, the $DEQ will fail
                      0216       ; with SS$_IVLOCKID, which is expected.
                      0216       ;-
                      0216
            00    E5  0216               BBCC    #RLB$V_TIMER_INPROG,-   ; Continue if no $SETIMR outstanding.
      0B 04 A3        0218                       RLB$W_FLAGS2(R3),105$
                      021B               $CANTIM_S -                     ; Cancel timer request.
                      021B                       REQIDT=R3
                      0226
 50   14 A3   3C      0226       105$:   MOVZWL  RLB$W_STATUS(R3),R0     ; copy enq status
         06 50    E9  022A               BLBC    R0,110$                 ; branch on any error
                      022D               RMSSUC  OK_WAT                  ; success, but we waited
                  05  0232               RSB
                      0233                                               ;
       2C    50   D1  0233       110$:   CMPL    R0, #SS$_ABORT          ; Was the lock request cancelled?
             07   12  0236               BNEQ    120$                    ; No; go map error.
                      0238               RMSERR  TMO                     ; Primary status is timeout.
             08   11  023D               BRB     130$                    ; Join exit code.
                      023F       120$:   RMSERR  ENQ,R1                  ; default to ENQ for RMSMAPERR
          FDB9'   30  0244       125$:   BSBW    RM$MAPERR               ; note subroutine call, not branch!
           0189   30  0247       130$:   BSBW    RESET_RLB               ; clear rlb, since we didn't get a lock
                  05  024A               RSB                             ; go return
                      024B
                      024B       NOBDB:  RMSPBUG FTL$_NOCURBDB           ; there should be a current BDB
                      0252          .SBTTL  SCAN
                      0252
                      0252  ;++
                      0252  ; SCAN
                      0252  ;       - scan the rlb for the requested record (rfa <>0)
                      0252  ;       - scan for first record locked by caller if rfa = 0
                      0252  ;       - report staus of scan
                      0252  ;
                      0252  ;
                      0252  ; Calling sequence:
                      0252  ;               bsbb    scan
                      0252  ;
                      0252  ;
                      0252  ; Input Parameters:
                      0252  ;
                      0252  ;       r11     impure area address
                      0252  ;       r10     ifab address
                      0252  ;       r9      irab address *** please note always irab ***
                      0252  ;       r8      rab/fab address
                      0252  ;
                      0252  ;       rfa <> 0 :      scan for record
                      0252  ;       r1      1'st and 2'nd word of record's rfa
                      0252  ;       r2      3'rd word of record's rfa
                      0252  ;                seq f.o.       offset (always positive value)
                      0252  ;                relative f.o.  always 0
                      0252  ;                index f.o.     low byte = record id
                      0252  ;
                      0252  ;       rfa = 0 :       scan for record locked by caller
                      0252  ;       r1      is zeroed (0)
                      0252  ;       r2      don't care
                      0252  ;
                      0252  ; Output Parameters:
                      0252  ;
                      0252  ;       r3 is rlb found on scan or 0 if none found in scan
                      0252  ;       r0      internal rms status code:
                      0252  ;
                      0252  ;               rms$_ok_alk&^xffff record was already locked by caller
                      0252  ;
                      0252  ;               rms$_rnl$^xffff record not locked by caller
                      0252  ;
                      0252  ; Side Effects:
                      0252  ;
                      0252  ;--
                      0252
                      0252 SCAN:                                          ;
  53   59   38   C1   0252          ADDL3   #IRB$L_RLB_LNK,R9,R3         ; get rlb header address into r3
            51   D5   0256          TSTL    R1                           ; owner scan
            30   13   0258          BEQL    SCAN_OWNER                   ; branch if yes
                      025A
                      025A  ;
                      025A  ; Scan for record match.
                      025A  ;
                      025A SCANLOOP:
                      025A          ASSUME  RLB$L_LNK EQ 0
       53   63   D0   025A          MOVL    (R3),R3                      ; get next rlb in list
            25   13   025D          BEQL    NOTFOUND                     ; branch if at end of list
   51    0C A3   D1   025F          CMPL    RLB$L_RFA0(R3),R1            ; compare vbn/rec#/start vbn
```

```
              F5   12  0263         BNEQ    SCANLOOP                ; branch if no match
                       0265
                       0265   ;
                       0265   ; Scans for sequential, relative, and index sequential file organization.
                       0265   ; Note:  For relative file organization only need to match record number.
                       0265   ;
                       0265
                       0265         CASE    TYPE=B,SRC=IFB$B_ORGCASE(R10),DISPLIST=<SCANSEQ,FOUND,SCANIDX>
                       0270
                       0270   ;
                       0270   ; Scan for sequential file organization.
                       0270   ;
                       0270   SCANSEQ:
    52     06 A3   B1  0270         CMPW    RLB$W_RFA4(R3),R2       ; compare offset
              08   13  0274         BEQL    FOUND                   ; branch if match
              E2   11  0276         BRB     SCANLOOP                ; otherwise, loop back for next
                       0278
                       0278   ;
                       0278   ; Scan for indexed file organization.
                       0278   ;
                       0278   SCANIDX:
    52     06 A3   B1  0278         CMPW    RLB$W_ID(R3),R2         ; compare id
              DC   12  027C         BNEQ    SCANLOOP                ; branch if no match
                       027E                                         ;
                       027E   ;
                       027E   ; Match has been found - report status.
                       027E   ;
                       027E   FOUND:                                ; ref tag
                       027E         RMSSUC  OK_ALK                  ; OK_ALK if caller already owns lock
                   05  0283         RSB                             ; and return
                       0284
                       0284   ;
                       0284   ; No match found.
                       0284   ;
                       0284   NOTFOUND:
                       0284         RMSERR  RNL                     ; set status and return
                   05  0289         RSB                             ;
                       028A
                       028A   ;
                       028A   ; Scan rlb list for owner match.
                       028A   ;
                       028A   SCAN_OWNER:
                       028A         ASSUME  RLB$L_LNK EQ 0
       53  63      D0  028A         MOVL    (R3),R3                 ; get next rlb in list
              F5   13  028D         BEQL    NOTFOUND                ; branch if at end of list
           10 A3   D5  028F         TSTL    RLB$L_OWNER(R3)         ; is RLB in use?
              F6   13  0292         BEQL    SCAN_OWNER              ; branch if not
                       0294         RMSSUC  OK_ALK                  ; set status and return
                   05  0299         RSB
```

```
                        029A   766         .SBTTL  RUSCAN
                        029A   767
                        029A   768 ;++
                        029A   769 ; RUSCAN
                        029A   770 ;       Look for RLB in RU lock list
                        029A   771 ;
                        029A   772 ;
                        029A   773 ; Calling sequence:
                        029A   774 ;               bsbb    ruscan
                        029A   775 ;
                        029A   776 ;
                        029A   777 ; Input Parameters:
                        029A   778 ;
                        029A   779 ;       r11     impure area address
                        029A   780 ;       r10     ifab address
                        029A   781 ;       r9      irab address *** please note always irab ***
                        029A   782 ;       r8      rab/fab address
                        029A   783 ;       r1      1'st and 2'nd word of record's rfa
                        029A   784 ;       r2      3'rd word of record's rfa
                        029A   785 ;                seq f.o.        offset (always positive value)
                        029A   786 ;                relative f.o.  always 0
                        029A   787 ;                index f.o.     low byte = record id
                        029A   788 ;
                        029A   789 ; Output Parameters:
                        029A   790 ;
                        029A   791 ;       r3 is rlb found on scan
                        029A   792 ;       r0
                        029A   793 ;               RMSSUC  - RLB found
                        029A   794 ;               0       - RLB not found
                        029A   795 ;               any error code - Returned from $ENQ service.
                        029A   796 ;
                        029A   797 ; Side Effects:
                        029A   798 ;
                        029A   799 ;--
                        029A   800
        0244     31     029A   801 ERRS:   BRW     ERRENQ
                        029D   802
                        029D   803
                        029D   804 RUSCAN:
          06     BB     029D   805         PUSHR   #^M<R1,R2>                              ; save R1,R2
        008D     30     029F   806         BSBW    FLB_SCAN                                ; find FLB for this file
       7F 50     E9     02A2   807         BLBC    R0,T10$                                 ; error if none
          50     D4     02A5   808         CLRL    R0                                      ; assume failure
  52   04 A1     DE     02A7   809         MOVAL   FLB$L_RLB_LNK(R1),R2                    ; get first RLB
                        02AB   810
     52   62     D0     02AB   811 10$:    MOVL    (R2),R2                                 ; next RLB into R2
          74     13     02AE   812         BEQL    110$                                    ; error if none
34 A9   10 A2    D1     02B0   813         CMPL    RLB$L_OWNER(R2),IRB$L_IDENT(R9)         ; locked by this stream?
          F4     12     02B5   814         BNEQ    10$                                     ; branch if not
  6E   0C A2     D1     02B7   815         CMPL    RLB$L_RFA0(R2),(SP)                     ; compare VBN/REC#/start vbn
          EE     12     02BB   816         BNEQ    10$                                     ; branch if no match
                        02BD   817
                        02BD   818 ;
                        02BD   819 ; Scans for sequential, relative, and index sequential file organization.
                        02BD   820 ; Note: For relative fiel organization only need to match record number.
                        02BD   821 ;
                        02BD   822
                        02BD                 CASE    TYPE=B,SRC=IFB$B_ORGCASE(R10),DISPLIST=<30$,50$,40$>
                        02C8
                        02C8      ;
                        02C8      ; Scan for sequental file organization.
                        02C8      ;
                        02C8
     04 AE    06 A2  B1 02C8      30$:       CMPW    RLB$W_RFA4(R2),4(SP)                        ; compare offset
                 DC  12 02CD                 BNEQ    10$                                         ; branch if no match
                 07  11 02CF                 BRB     50$                                         ; match found
                        02D1
                        02D1      ;
                        02D1      ; Scan for indexed file organization
                        02D1      ;
                        02D1
     04 AE    06 A2  B1 02D1      40$:       CMPW    RLB$W_ID(R2),4(SP)                          ; compare id
                 D3  12 02D6                 BNEQ    10$                                         ; branch if no match
                        02D8
                        02D8      ;
                        02D8      ; RLB found, remove from RU RLB list, insert in IRB RLB list
                        02D8      ;
                        02D8
               00A8  30 02D8      50$:       BSBW    GET_RLB                                     ; get a RLB
                 3E  BB 02DB                 PUSHR   #^M<R1,R2,R3,R4,R5>                         ; save registers
  04 A3   04 A2  18  28 02DD                 MOVC3   #RLB$C_BLN-4,4(R2),4(R3)                    ; copy saved RLB
     53    04 AE     D0 02E3                 MOVL    4(SP),R3                                    ; get saved RLB address
               00E9  30 02E7                 BSBW    RESET_RLB                                   ; clear it out
                 3E  BA 02EA                 POPR    #^M<R1,R2,R3,R4,R5>                         ; restore registers
     0B A3       21  CA 02EC                 BICL    #<RLB$M_WAIT!RLB$M_LV2>,RLB$B_FLAGS(R3)     ; clear WAIT, LV2
     05 68       2C  E1 02F0                 BBC     #RAB$V_LV2+ROP,(R8),60$                     ; branch if no LV2 now
                        02F4                 SSB     #RLB$V_LV2,RLB$B_FLAGS(R3)                  ; set LV2
                        02F9      ;
                        02F9      ; Find out if we have the lock in the right mode
                        02F9      ;
                        02F9
     05 68       22  E1 02F9      60$:       BBC     #RAB$V_REA+ROP,(R8),70$                     ; branch if not REA lock
  0E 0B A3       03  E3 02FD                 BBCS    #RLB$V_PR,RLB$B_FLAGS(R3),90$               ; branch if not already PR
                        0302
     05 68       33  E1 0302      70$:       BBC     #RAB$V_RLK+ROP,(R8),80$                     ; branch if not RLK lock
  05 0B A3       02  E3 0306                 BBCS    #RLB$V_PW,RLB$B_FLAGS(R3),90$               ; branch if not alread PW
                        030B      80$:       RMSSUC                                              ; set success
                 14  11 030E                 BRB     110$                                        ; lock in correct mode, get
                        0310
                        0310      ;
                        0310      ; We have to change the mode of the lock
                        0310      ;
                        0310
     05 68       31  E1 0310      90$:       BBC     #RAB$V_WAT+ROP,(R8),100$                    ; branch if not wait
                        0314                 SSB     #RLB$V_WAIT,RLB$B_FLAGS(R3)                 ; set wait
                        0319      100$:      SSB     #RLB$V_CONV,RLB$B_FLAGS(R3)                 ; set convert
               FE05  30 031E                 BSBW    DO_ENQ                                      ; go do ENQ
               FDF1  30 0321                 BSBW    RRC                                         ; set codes
                        0324
                 06  BA 0324      110$:      POPR    #^M<R1,R2>                                  ; restore R1,R2
              05 50  E9 0326                 BLBC    R0,120$                                     ; get out on error
                        0329                 RMSSUC  OK_RULK                                     ; alternate success code
                        032E
                     05 032E      120$:      RSB
```

```
                                032F   881          .SBTTL  FLB_SCAN
                                032F   882  ;++
                                032F   883  ; FLB_SCAN
                                032F   884  ;        Search for an FLB which matches the current IFB address
                                032F   885  ;--
                                032F   886
                                032F   887  FLB_SCAN:
                    50    D4    032F   888          CLRL    R0                                          ; assume failure
   51    00000000'9F      DE    0331   889          MOVAL   @#PIO$GL_RULOCK,R1                          ; get FLB list
                                0338   890
              51    61    D0    0338   891  10$:    MOVL    (R1),R1                                     ; get next FLB
                    0E    13    033B   892          BEQL    20$                                         ; get out if none
         0C A1      5A    D1    033D   893          CMPL    R10,FLB$L_IFB_PTR(R1)                       ; see if this IFB
                    F5    12    0341   894          BNEQ    10$                                         ; branch if not
              10    A1    D5    0343   895          TSTL    FLB$L_LOCK_ID(R1)                           ; saved file lock here?
                    F0    12    0346   896          BNEQ    10$                                         ; skip it if so.
                                0348   897          RMSSUC
                                034B   898
                          05    034B   899  20$:    RSB
```

```
                  034C          .SBTTL  PRSCAN
                  034C
                  034C   ;++
                  034C   ; PRSCAN
                  034C   ;          Look for RLB in RU lock list, disregarding stream, not returning lock
                  034C   ;
                  034C   ;
                  034C   ; Calling sequence:
                  034C   ;                  bsbb     prscan
                  034C   ;
                  034C   ;
                  034C   ; Input Parameters:
                  034C   ;
                  034C   ;          r11      impure area address
                  034C   ;          r10      ifab address
                  034C   ;          r9       irab address *** please note always irab ***
                  034C   ;          r8       rab/fab address
                  034C   ;          r1       1'st and 2'nd word of record's rfa
                  034C   ;          r2       3'rd word of record's rfa
                  034C   ;                    seq f.o.          offset (always positive value)
                  034C   ;                    relative f.o.  always 0
                  034C   ;                    index f.o.        low byte = record id
                  034C   ;
                  034C   ; Output Parameters:
                  034C   ;
                  034C   ;          r0
                  034C   ;                   RMSSUC   - RLB found
                  034C   ;                   0        - RLB not found
                  034C   ;
                  034C   ; Side Effects:
                  034C   ;
                  034C   ;--
         04   BB  034C   PRSCAN: PUSHR     #^M<R2>                               ; save R2
         DF   10  034E           BSBB      FLB_SCAN                              ; get FLB address
      2D 50   E9  0350           BLBC      R0,60$                                ; get out if none
52    04 A1   DE  0353           MOVAL     FLB$L_RLB_LNK(R1),R2                  ; get pointer to RLBs
         50   D4  0357           CLRL      R0                                    ; assume failure
                  0359
   52    62   D0  0359   10$:    MOVL      (R2),R2                               ; next RLB into R2
         22   13  035C           BEQL      60$                                   ; get out if none
51    0C A2   D1  035E           CMPL      RLB$L_RFA0(R2),R1                     ; compare VBN/REC#/start vbn
         F5   12  0362           BNEQ      10$                                   ; branch if no match
                  0364
                  0364   ;
                  0364   ; Scans for sequential, relative, and index sequential file organization.
                  0364   ; Note: for relative file organization only need to match record number.
                  0364   ;
                  0364
                  0364           CASE      TYPE=B,SRC=IFB$B_ORGCASE(R10),DISPLIST=<30$,50$,40$>
                  036F
                  036F   ;
                  036F   ; Scan for sequential file organization
                  036F   ;
                  036F
6E    06 A2   B1  036F   30$:    CMPW      RLB$W_RFA4(R2),(SP)                   ; compare offset
         E4   12  0373           BNEQ      10$                                   ; branch if no match
         06   11  0375           BRB       50$                                   ; match found
```

```
                     0377   958
                     0377   959 ;
                     0377   960 ; Scan for indexed file organization
                     0377   961 ;
                     0377   962
  6E   06 A2   B1    0377   963 40$:    CMPW    RLB$W_ID(R2),(SP)                   ; compare id
          DC   12    037B   964         BNEQ    10$                                 ; branch if no match
                     037D   965
                     037D   966
                     037D   967 50$:    RMSSUC
                     0380   968
          04   BA    0380   969 60$:    POPR    #^M<R2>
               05    0382   970         RSB
```

```
                 0383            .SBTTL  GET_RLB AND RESET_RLB
                 0383       ;++
                 0383       ; GET_RLB        - find an available rlb, if none available allocate one
                 0383       ; RESET_RLB      - clear the RLB and indicate its free
                 0383       ;
                 0383       ; Calling Sequence:
                 0383       ;                       bsbb    get_rlb
                 0383       ;                       bsbb    reset_rlb
                 0383       ;
                 0383       ; Input Parameters:
                 0383       ;
                 0383       ;       get_rlb:
                 0383       ;       r10     ifab address
                 0383       ;       r9      irab address
                 0383       ;       r1      1'st and 2'nd word of record's rfa
                 0383       ;       r2      3'rd word of record's rfa
                 0383       ;                seq f.o.        offset (always positive value)
                 0383       ;                relative f.o.   always 0
                 0383       ;                index f.o.      low byte = record id
                 0383       ;
                 0383       ; Output Parameters:
                 0383       ;
                 0383       ;       get_rlb:
                 C383       ;       r3 points to RLB if success, else zero
                 0383       ;
                 0383       ;       r0 internal RMS status code:
                 0383       ;               DME - couldn't allocate an RLB
                 0383       ;               SUC - r3 points to RLB
                 0383       ;
                 0383       ; Side Effects:
                 0383       ;
                 0383       ;       If success, RLB owner and RFA fields initialized.
                 0383       ;--
                 0383
                 0383       ;
                 0383       ; Record is not in local list of locked records, so scan the rlb list for
                 0383       ; an available rlb.
                 0383       ;
                 0383
                 0383       GET_RLB:                                        ; find an rlb
 53    59   38   C1  0383           ADDL3   #IRB$L_RLB_LNK,R9,R3    ; get rlb header address
                 0387               ASSUME  RLB$L_LNK EQ 0
       53   63   D0  0387   10$:    MOVL    (R3),R3                 ; get next rlb in list
            07   13  038A           BEQL    20$                     ; if eql end of list, go allocate one
         10 A3   D5  038C           TSTL    RLB$L_OWNER(R3)         ; is rlb available
            F6   12  038F           BNEQ    10$                     ; loop back for next if not
            29   11  0391           BRB     30$                     ; success, r3 points to rlb
                 0393
                 0393       ;
                 0393       ; No available rlb so we must allocate a new one.
                 0393       ;
            16   BB  0393   20$:    PUSHR   #^M<R1,R2,R4>           ; save registers
       51   5A   D0  0395           MOVL    R10,R1                  ; set addr in page = ifab
       52   07   9A  0398           MOVZBL  #RLB$C_BLN/4,R2         ; set # of long words
         FC62'   30  039B           BSBW    RM$GETBLK               ; get rlb block and fill in length
       53   51   D0  039E           MOVL    R1,R3                   ; copy address if any
            16   BA  03A1           POPR    #^M<R1,R2,R4>           ; restore registers
```

```
          27 50   E9 03A3                 BLBC    R0,ERRDME                 ; if we failed then exit
                     03A6                 ASSUME  RLB$B_BLN EQ RLB$B_BID+1
        070E 8F   B0 03A6                 MOVW    #RLB$C_BID+<RLB$C_BLN@6>,-
          08 A3      03AA                         RLB$B_BID(R3)             ; set block id code
        50    53  D0 03AC                 MOVL    R3,R0                     ; save new rlb address
   53   59    38  C1 03AF                 ADDL3   #IRB$L_RLB_LNK,R9,R3      ; get rlb header address
                     03B3                 ASSUME  RLB$L_LNK EQ 0
        60    63  D0 03B3                 MOVL    (R3),(R0)                 ; set ptr to next in new rlb
        63    50  D0 03B6                 MOVL    R0,(R3)                   ; put new rlb at front of list
        53    50  D0 03B9                 MOVL    R0,R3                     ; restore new rlb address
                     03BC       30$:                                        ; initialize RLB
                     03BC       SETOWNRFA:                                  ; can be called here by QUERY_LCK
          34 A9   D0 03BC                 MOVL    IRB$L_IDENT(R9),-         ;
          10 A3      03BF                         RLB$L_OWNER(R3)           ; set owner isi
     0C A3    51  D0 03C1                 MOVL    R1,RLB$L_RFA0(R3)         ; set records rfa in rlb
     06 A3    52  B0 03C5                 MOVW    R2,RLB$W_RFA4(R3)         ;
                     03C9                 RMSSUC                            ; indicate success
                  05 03CC                 RSB                               ; and return
                     03CD
                     03CD       ;
                     03CD       ; error allocating rlb
                     03CD       ;
                     03CD       ERRDME: RMSERR  DME                         ; no dynamic memory
                  05 03D2                 RSB                               ; return to caller
                     03D3
                     03D3
                     03D3       ;++
                     03D3       ; RESET_RLB
                     03D3       ;
                     03D3       ; Indicate the RLB is free, and clean it up. Called from UNLOCK and errors
                     03D3       ; on LOCK.
                     03D3       ;
                     03D3       ;       r0 must be preserved by this routine
                     03D3       ;       r3 points to the RLB
                     03D3       ;
                     03D3       ; Note that RLB$B_TMO is not cleared, since it is only meaningful when
                     03D3       ; RLB$V_TMO is set in RLB$B_FLAGS, which is cleared here.
                     03D3       ;--
                     03D3
                     03D3                 ASSUME  <RLB$L_RFA0+4> EQ RLB$L_OWNER
                     03D3       RESET_RLB:
          04 A3   D4 03D3                 CLRL    RLB$L_MISC(R3)            ; Clears RLB$W_FLAGS2.
          0C A3   7C 03D6                 CLRQ    RLB$L_RFA0(R3)            ;
          0B A3   94 03D9                 CLRB    RLB$B_FLAGS(R3)           ;
                  05 03DC                 RSB                               ; return to caller
```

```
                     03DD          .SBTTL  RM$UNLOCK AND RI $UNLOCKALL
                     03DD
                     03DD    ;++
                     03DD    ; RM$UNLOCK
                     03DD    ; RM$UNLOCKALL
                     03DD    ;
                     03DD    ;       Deletes entries in the record lock list
                     03DD    ;
                     03DD    ; RM$UNLOCK_HARD
                     03DD    ;
                     03DD    ;       Deletes an entry in the record lock list, but maps a REA lock held by
                     03DD    ;       the caller to RNL so a writer holding a REA lock does not attempt an
                     03DD    ;       update or delete.
                     03DD    ;
                     03DD    ; Calling sequence:
                     03DD    ;               bsbw    rm$unlock
                     03DD    ;               bsbw    rm$unlockall
                     03DD    ;               bsbw    rm$unlock_hard
                     03DD    ;
                     03DD    ;
                     03DD    ; Input Parameters:
                     03DD    ;
                     03DD    ;       r11     impure area address
                     03DD    ;       r10     ifab (shared ifab) address
                     03DD    ;       r9      irab address *** please note always irab ***
                     03DD    ;       r8      rab/fab address
                     03DD    ;
                     03DD    ;       rfa <> 0 :      unlock record
                     03DD    ;       r1      1'st and 2'nd word of record's rfa
                     03DD    ;       r2      3'rd word of record's rfa
                     03DD    ;                seq f.o.       offset (always positive value)
                     03DD    ;                relative f.o.  always 0
                     03DD    ;                index f.o.     low byte = record id
                     03DD    ;
                     03DD    ;       rm$unlockall entry
                     03DD    ;       r1,r2 don't care
                     03DD    ;
                     03DD    ; Output Parameters:
                     03DD    ;
                     03DD    ;       r3 is destroyed
                     03DD    ;
                     03DD    ;       r0      internal rms status code
                     03DD    ;               rms$_suc&^xffff record(s) unlocked
                     03DD    ;               rms$_rnl&^xffff record was not locked
                     03DD    ;                               or no record was locked (unlock all call)
                     03DD    ;       rm$unlockall:
                     03DD    ;       the irb$v_unlock_rp  irab bookeeping bit is cleared
                     03DD    ;       r1 is zeroed
                     03DD    ;
                     03DD    ;
                     03DD    ; Side Effects:
                     03DD    ;
                     03DD    ;--
                     03DD
                     03DD    RM$UNLOCK_HARD::
          FE72   30  03DD          BSBW    SCAN                    ; find record
50   8039 8F     B1  03E0          CMPW    #<RMS$_OK_ALK&^XFFFF>,R0; caller locked record?
```

```
               0C  12  03E5  1132              BNEQ    10$                       ; if neq no, return RNL error
               03  E1  03E7  1133              BBC     #RLB$V_PR,-               ; did caller lock record REA?
         18 0B A3      03E9  1134                      RLB$B_FLAGS(R3),DEQUE     ; no, continue usual path
               16  10  03EC  1135              BSBB    DEQUE                     ; yes, go unlock the sucker and...
                       03EE  1136              RMSERR  RNL                       ; return RNL so no update is attempted
                   05  03F3  1137 10$:         RSB                               ; return to caller
                       03F4  1138
                       03F4  1139 DEQUE_QUERY:                                   ; called here from QUERY
      25 6A    33  E1  03F4  1140              BBC     #IFB$V_NORECLK,(R10),DEQ ; do a deq if record locking
               36  11  03F8  1141              BRB     DEQ_RS                    ; go release RLB
                       03FA  1142
                       03FA  1143 RM$UNLOCK::                                    ;
                       03FA  1144 UNLOCK:                                        ; ref tag
             FE55  30  03FA  1145              BSBW    SCAN                      ; scan for record
   50    8039 8F   B1  03FD  1146              CMPW    #<RMS$_OK_ALK&^XFFFF>,R0; did we find a locked record for stream
               32  12  0402  1147              BNEQ    NOTLOCK                   ; branch if no
                       0404  1148
                       0404  1149 ;++
                       0404  1150 ;
                       0404  1151 ; Perform the $DEQ_S
                       0404  1152 ;
                       0404  1153 ;--
                       0404  1154
               06  E0  0404  1155 DEQUE:  BBS      #RLB$V_FAKE,-             ; if 'fake' RLB then maybe RUSAVE
         04 0B A3      0406  1156                      RLB$B_FLAGS(R3),10$
               33  E0  0409  1157              BBS     #IFB$V_NORECLK,-          ; dont do a deq if no record locking
            23 6A      040B  1158                      (R10),DEQ_RS              ;
                       040D  1159                                                ;
               02  E1  040D  1160 10$:    BBC      #IFB$V_RUP,-
       0A 00A2 CA      040F  1161                      IFB$B_JNLFLG2(R10),DEQ    ; branch if not in RU
               05  E1  0413  1162              BBC     #RLB$V_LV2,-
         24 0B A3      0415  1163                      RLB$B_FLAGS(R3),RUSAVE    ; save lock if not Level 2
               03  E1  0418  1164              BBC     #RLB$V_PR,-               ; if level 2 save all but
         1F 0B A3      041A  1165                      RLB$B_FLAGS(R3),RUSAVE    ;      PR locks.
    0E 0B A3   06  E0  041D  1166 DEQ:    BBS      #RLB$V_FAKE,RLB$B_FLAGS(R3),DEQ_RS ; branch if fake RLB
                       0422  1167              $DEQ_S  LKID=RLB$L_LOCK_ID(R3)    ; lock id of lock to unlock
                       0430  1168                                                ; ignore errors...
                       0430  1169                                                ;
               A1  10  0430  1170 DEQ_RS: BSBB     RESET_RLB                 ; free the rlb
                       0432  1171              RMSSUC                            ; say success
                   05  0435  1172              RSB                               ; and return
                       0436  1173                                                ;
                       0436  1174 NOTLOCK:                                       ;
                       0436  1175              RMSERR  RNL                       ; say record not locked
                   05  043B  1176              RSB                               ; and exit
                       043C  1177
                       043C  1178 ;
                       043C  1179 ; Save locks given up in Recovery Units
                       043C  1180 ;
                       043C  1181
                       043C  1182              ASSUME  <RLB$C_BLN+1> GT FLB$C_BLN
                       043C  1183
               3E  BB  043C  1184 RUSAVE: PUSHR    #^M<R1,R2,R3,R4,R5>       ; save registers
             FEEE  30  043E  1185              BSBW    FLB_SCAN                  ; see if there is already an FLB
            1B 50  E8  0441  1186              BLBS    R0,70$                    ; branch if so
               4E  10  0444  1187              BSBB    ALOCPBLK                  ; get an FLB
            3F 50  E9  0446  1188              BLBC    R0,50$                    ; get out on error
   61    00000000'9F    D0  0449  1189            MOVL    @#PIO$GL_RULOCK,(R1)     ; set successor to first FLB
   00000000'9F    51    D0  0450  1190            MOVL    R1,@#PIO$GL_RULOCK       ; set new FLB as first FLB
          08 A1   17    90  0457  1191            MOVB    #FLB$C_BID,FLB$B_BID(R1)         ; set block id in FLB
          0C A1   5A    D0  045B  1192            MOVL    R10,FLB$L_IFB_PTR(R1)    ; set IFAB address in FLB
                            045F  1193
             52   51    D0  045F  1194 10$:       MOVL    R1,R2                    ; save FLB address
          51   04 A2    DE  0462  1195            MOVAL   FLB$L_RLB_LNK(R2),R1     ; get RLB pointer
                            0466  1196
             51   61    D0  0466  1197 20$:       MOVL    (R1),R1                  ; get RLB
                  07    13  0469  1198            BEQL    30$                      ; branch if none
               10 A1    D5  046B  1199            TSTL    RLB$L_OWNER(R1)          ; is RLB available
                  F6    12  046E  1200            BNEQ    20$                      ; branch if not
                  0D    11  0470  1201            BRB     40$                      ; go use it otherwise
                            0472  1202
                  20    10  0472  1203 30$:       BSBB    ALOCPBLK                 ; get an RLB
               11 50    E9  0474  1204            BLBC    R0,50$                   ; get out on error
          61   04 A2    D0  0477  1205            MOVL    FLB$L_RLB_LNK(R2),(R1)   ; set successor to first RLB
          04 A2   51    D0  047B  1206            MOVL    R1,FLB$L_RLB_LNK(R2)     ; set new RLB as first RLB
                            047F  1207
    04 A1    04 A3   18    28  047F  1208 40$:       MOVC3   #RLB$C_BLN-4,4(R3),4(R1); copy old RLB
                            0485  1209            RMSSUC
                            0488  1210
                  3E    BA  0488  1211 50$:       POPR    #^M<R1,R2,R3,R4,R5>
                FF46    30  048A  1212            BSBW    RESET_RLB                ; remove lock from IFB RLB list
               03 50    E8  048D  1213            BLBS    R0,60$                   ; return on success
                FF3A    31  0490  1214            BRW     ERRDME                   ; only error possible is DME
                        05  0493  1215 60$:       RSB
                            0494  1216
                            0494  1217 ;
                            0494  1218 ; ALOCPBLK - Allocate a block for the RULOCK list
                            0494  1219 ;
                            0494  1220
                            0494  1221 ALOCPRLK:
             083C 8F    BB  0494  1222            PUSHR   #^M<R2,R3,R4,R5,R11>     ; save registers
   5B    00000000'9F    DE  0498  1223            MOVAL   @#PIO$GW_PIOIMPA,R11     ; PIO free list header
             51   5B    D0  049F  1224            MOVL    R11,R1
             52   07    9A  04A2  1225            MOVZBL  #RLB$C_BLN/4,R2          ; set # of long words
               FB58'    30  04A5  1226            BSBW    RM$GETBLK                ; get block
             083C 8F    BA  04A8  1227            POPR    #^M<R2,R3,R4,R5,R11>     ; restore registers
                        05  04AC  1228            RSB
                            04AD  1229
                            04AD  1230 ;
                            04AD  1231 ; DEAPBLK - Deallocate a RULOCK block
                            04AD  1232 ;
                            04AD  1233
                            04AD  1234 DEAPBLK:
             083C 8F    BB  04AD  1235            PUSHR   #^M<R2,R3,R4,R5,R11>
   5B    00000000'9F    DE  04B1  1236            MOVAL   @#PIO$GW_PIOIMPA,R11     ; PIO free list header
             53   5B    D0  04B8  1237            MOVL    R11,R3
             54   51    D0  04BB  1238            MOVL    R1,R4
               FB3F'    30  04BE  1239            BSBW    RM$RETBLK                ; return space
             083C 8F    BA  04C1  1240            POPR    #^M<R2,R3,R4,R5,R11>
                        05  04C5  1241            RSB
                            04C6  1242
                            04C6  1243 ;
                            04C6  1244 ; Unlock all records for the caller.
                            04C6  1245 ;
                04C6  1246
                04C6  1247 RM$UNLOCKALL::                                 ;
                04C6  1248          CSB     #IRB$V_UNLOCK_RP,(R9)         ; rp will be unlocked so note that
                04CA  1249                                                ; it was done
      51    D4  04CA  1250          CLRL    R1                            ; flag owner scan
    FF2B    30  04CC  1251          BSBW    UNLOCK                        ; unlock first record
   0C 50    E9  04CF  1252          BLBC    R0,NTLK                       ; if we failed then exit
      51    D4  04D2  1253 10$:     CLRL    R1                            ; re-flag owner scan, since DEQ blew R1
    FF23    30  04D4  1254          BSBW    UNLOCK                        ; unlock next record
   F8 50    E8  04D7  1255          BLBS    R0,10$                        ; loop back if success
                04DA  1256          RMSSUC                                ; exit with success
            05  04DD  1257          RSB
                04DE  1258
    FF55    31  04DE  1259 NTLK:    BRW     NOTLOCK
                04E1  1260                                                ;
                04E1  1261 ERRENQ: RMSPBUG FTL$_ENQDEQFAIL                ; the deq failed
```

```
                                 04E8  1263         .SBTTL  RMSSAVE_FL
                                 04E8  1264 ;++
                                 04E8  1265 ; RMSSAVE_FL - Save the file lock in the RULOCK list
                                 04E8  1266 ;
                                 04E8  1267 ; Calling sequence:
                                 04E8  1268 ;               BSBW    RMSSAVE_FL
                                 04E8  1269 ;
                                 04E8  1270 ; Input Parameters:
                                 04E8  1271 ;       R4      -       SFSB address
                                 04E8  1272 ;       R9      -       IFAB address
                                 04E8  1273 ;
                                 04E8  1274 ; Output Parameters:
                                 04E8  1275 ;       R1      -       Destroyed
                                 04E8  1276 ;
                                 04E8  1277 ; Side Effects:
                                 04E8  1278 ;       None.
                                 04E8  1279 ;
                                 04E8  1280 ;--
                                 04E8  1281
                                 04E8  1282 RMSSAVE_FL::
                        5A   DD  04E8  1283         PUSHL   R10                             ; save R10
                   5A   59   DO  04EA  1284         MOVL    R9,R10                          ; move IFB address for FLB_SCAN
                      FE3F   30  04ED  1285         BSBW    FLB_SCAN                        ; see if there is an FLB
                     1B 50   E8  04F0  1286         BLBS    R0,10$                          ; branch if one
                        9F   10  04F3  1287         BSBB    ALOCPBLK                        ; get an FLB
                     1B 50   E9  04F5  1288         BLBC    R0,20$                          ; get out on error
           61   00000000'9F  D0  04F8  1289         MOVL    @#PIO$GL_RULOCK,(R1)            ; set successor to first FLB
           00000000'9F  51   D0  04FF  1290         MOVL    R1,@#PIO$GL_RULOCK              ; set new FLB as first FLB
                08 A1   17   90  0506  1291         MOVB    #FLB$C_BID,FLB$B_BID(R1)        ; set block id in FLB
                0C A1   59   D0  050A  1292         MOVL    R9,FLB$L_IFB_PTR(R1)            ; set IFAB address in FLB
                                 050E  1293
          10 A1     30 A4    D0  050E  1294 10$:    MOVL    SFSB$L_LOCK_ID(R4),FLB$L_LOCK_ID(R1) ; save lock
                                 0513  1295
                       5A 8ED0   0513  1296 20$:    POPL    R10                             ; restore R10
                             05  0516  1297         RSB
                        0517          .SBTTL  RM$RU_UNLOCK
                        0517  
                        0517  ;++
                        0517  ; RM$RU_UNLOCK - Unlocks all locks held for recovery unit
                        0517  ;
                        0517  ; Calling sequence:
                        0517  ;               BSBW    RM$RU_UNLOCK
                        0517  ;
                        0517  ; Input Parameters:
                        0517  ;       None.
                        0517  ;
                        0517  ; Ouput Parameters:
                        0517  ;       R0-R5   Destroyed
                        0517  ;
                        0517  ; Side Effects:
                        0517  ;       None.
                        0517  ;
                        0517  ;--
                        0517  
                        0517  RM$RU_UNLOCK::
              5B  DD    0517          PUSHL   R11                             ; save impure area address
5B   00000000'9F  DE    0519          MOVAL   @#PIO$GW_PIOIMPA,R11            ; point to process I/O set
53   00000000'9F  D0    0520          MOVL    @#PIO$GL_RULOCK,R3              ; get first FLB
                        0527  
              41  13    0527  10$:    BEQL    60$                             ; branch if none
     54    04 A3  D0    0529          MOVL    FLB$L_RLB_LNK(R3),R4            ; get RLB address
                        052D  
              1B  13    052D  20$:    BEQL    40$                             ; branch if at end of list
                        052F          $DEQ_S  LKID=RLB$L_LOCK_ID(R4)          ; deque record lock
              64  DD    053D          PUSHL   (R4)                            ; save next RLB address
         51   54  D0    053F          MOVL    R4,R1                           ; return RLB
            FF68  30    0542          BSBW    DEAPBLK
         54   8E  D0    0545          MOVL    (SP)+,R4                        ; get next RLB
              E3  11    0548          BRB     20$                             ; go process next RLB
                        054A  
           10 A3  D5    054A  40$:    TSTL    FLB$L_LOCK_ID(R3)               ; was a file lock saved?
              0E  13    054D          BEQL    50$                             ; branch if not
                        054F          $DEQ_S  LKID=FLB$L_LOCK_ID(R3)          ; deque file lock
              63  DD    055D  50$:    PUSHL   (R3)                            ; save next FLB address
         51   53  D0    055F          MOVL    R3,R1                           ; return FLB
            FF48  30    0562          BSBW    DEAPBLK
         53   8E  D0    0565          MOVL    (SP)+,R3                        ; get next FLB address
              BD  11    0568          BRB     10$                             ; go process it
                        056A  
     00000000'9F  D4    056A  60$:    CLRL    @#PIO$GL_RULOCK                 ; clear list header
         5B   8E  D0    0570          MOVL    (SP)+,R11                       ; restore R11
                  05    0573          RSB
                        0574  
                        0574          .END
```

```
$$.PSECT_EP                     = 00000000                  LCK$K_CRMODE                    = 00000001
$$ARGS                          = 0000000B                  LCK$K_EXMODE                    = 00000005
$$RMSTEST                       = 0000001A                  LCK$K_PRMODE                    = 00000003
$$RMS_PBUGCHK                   = 00000010                  LCK$K_PWMODE                    = 00000004
$$RMS_TBUGCHK                   = 00000008                  LCK$M_CONVERT                   = 00000002
$$RMS_UMODE                     = 00000004                  LCK$M_NOQUEUE                   = 00000004
$$T1                            = 00000001                  LCK$M_SYNCSTS                   = 00000008
ALOCPBLK                          00000494 R       01       LCK$M_SYSTEM                    = 00000010
DEAPBLK                           000004AD R       01       LCK$V_PROTECT                   = 00000008
DEQ                               0000041D R       01       LCK$V_RECOVER                   = 00000007
DEQUE                             00000404 R       01       NOBDB                             0000024B R       01
DEQUE_QUERY                       000003F4 R       01       NOTFOUND                          00000284 R       01
DEQ_RS                            00000430 R       01       NOTLOCK                           00000436 R       01
DO_ENQ                            00000126 R       01       NTLK                              000004DE R       01
ENQ$_ACMODE                     = 00000028                  PIO$A_TRACE                       ********   X     01
ENQ$_ASTADR                     = 0000001C                  PIO$GL_RULOCK                     ********   X     01
ENQ$_ASTPRM                     = 00000020                  PIO$GW_PIOIMPA                    ********   X     01
ENQ$_BLKAST                     = 00000024                  PRSCAN                            0000034C R       01
ENQ$_EFN                        = 00000004                  RAB$B_TMO                       = 0000001F
ENQ$_FLAGS                      = 00000010                  RAB$L_ROP                       = 00000004
ENQ$_LKMODE                     = 00000008                  RAB$V_LV2                       = 0000000C
ENQ$_LKSB                       = 0000000C                  RAB$V_REA                       = 00000002
ENQ$_NARGS                      = 0000000B                  RAB$V_RLK                       = 00000013
ENQ$_PARID                      = 00000018                  RAB$V_RRL                       = 00000003
ENQ$_PROT                       = 0000002C                  RAB$V_TMO                       = 00000019
ENQ$_RESNAM                     = 00000014                  RAB$V_WAT                       = 00000011
ERRDME                            000003CD R       01       RESET_RLB                         000003D3 R       01
ERRENQ                            000004E1 R       01       RLB$B_BID                       = 00000008
ERRS                              0000029A R       01       RLB$B_BLN                       = 00000009
FLB$B_BID                       = 00000008                  RLB$B_FLAGS                     = 0000000B
FLB$C_BID                       = 00000017                  RLB$B_TMO                       = 0000000A
FLB$C_BLN                       = 00000014                  RLB$C_BID                       = 0000000E
FLB$L_IFB_PTR                   = 0000000C                  RLB$C_BLN                       = 0000001C
FLB$L_LOCK_ID                   = 00000010                  RLB$L_LKSB                      = 00000014
FLB$L_RLB_LNK                   = 00000004                  RLB$L_LNK                       = 00000000
FLB_SCAN                          0000032F R       01       RLB$L_LOCK_ID                   = 00000018
FOUND                             0000027E R       01       RLB$L_MISC                      = 00000004
FTL$_ENQDEQFAIL                 = FFFFFFF2                  RLB$L_OWNER                     = 00000010
FTL$_NOCURBDB                   = FFFFFFF1                  RLB$L_RFA0                      = 0000000C
GET_RLB                           00000383 R       01       RLB$M_CR                        = 00000002
IFB$B_JNLFLG                    = 000000A0                  RLB$M_LV2                       = 00000020
IFB$B_JNLFLG2                   = 000000A2                  RLB$M_PR                        = 00000008
IFB$B_ORGCASE                   = 00000023                  RLB$M_WAIT                      = 00000001
IFB$B_RECVRFLGS                 = 000000A1                  RLB$V_CONV                      = 00000004
IFB$L_SFSB_PTR                  = 00000078                  RLB$V_CR                        = 00000001
IFB$M_ONLY_RU                   = 00000001                  RLB$V_FAKE                      = 00000006
IFB$M_RU                        = 00000002                  RLB$V_LV2                       = 00000005
IFB$V_NORECLK                   = 00000033                  RLB$V_PR                        = 00000003
IFB$V_RUP                       = 00000002                  RLB$V_PW                        = 00000002
IFB$V_RU_RECVR                  = 00000000                  RLB$V_TIMER_INPROG              = 00000000
IFB$V_RU_RLK                    = 00000003                  RLB$V_TMO                       = 00000007
IRB$B_EFN                       = 0000000B                  RLB$V_WAIT                      = 00000000
IRB$L_CURBDB                    = 00000020                  RLB$W_FLAGS2                    = 00000004
IRB$L_IDENT                     = 00000034                  RLB$W_ID                        = 00000006
IRB$L_RLB_LNK                   = 00000038                  RLB$W_RFA4                      = 00000006
IRB$V_NO_Q_WAIT                 = 00000038                  RLB$W_STATUS                    = 00000014
IRB$V_UNLOCK_RP                 = 0000002D                  RM$BUG                            ********   X     01
```

```
RMS$GETBLK                         ********    X    01
RMS$LOCK                           00000000 RG      01
RMS$MAPERR                         ********    X    01
RMS$QUERY_HARD                     00000070 RG      01
RMS$QUERY_LCK                      000000A1 RG      01
RMS$QUERY_PROC                     0000008B RG      01
RMS$RELEASE                        ********    X    01
RMS$RETBLK                         ********    X    01
RMS$RU_UNLOCK                      00000517 RG      01
RMS$SAVE_FL                        000004E8 RG      01
RMS$SETEFN                         ********    X    01
RMS$SET_LOCK_TMO                   ********    X    01
RMS$STALL                          ********    X    01
RMS$STALLAST                       ********    X    01
RMS$UNLOCK                         000003FA RG      01
RMS$UNLOCKALL                      000004C6 RG      01
RMS$UNLOCK_HARD                    000003DD RG      01
RMS$_DME                         = 000184D4
RMS$_ENQ                         = 0001C134
RMS$_OK_ALK                      = 00018039
RMS$_OK_RLK                      = 00018021
RMS$_OK_RRL                      = 00018029
RMS$_OK_RULK                     = 00018071
RMS$_OK_WAT                      = 00018061
RMS$_RLK                         = 000182AA
RMS$_RNL                         = 000181A0
RMS$_TMO                         = 000181B0
RMS$_TMR                         = 0001C16C
ROP                              = 00000020
RRL                                00000115 R       01
RUSAVE                             0000043C R       01
RUSCAN                             0000029D R       01
SCAN                               00000252 R       01
SCANIDX                            00000278 R       01
SCANLOOP                           0000025A R       01
SCANSEQ                            00000270 R       01
SCAN_OWNER                         0000028A R       01
SETOWNRFA                          000003BC R       01
SFSB$L_LOCK_ID                   = 00000030
SS$_ABORT                        = 0000002C
SS$_SYNCH                        = 00000689
SYS$CANTIM                         ********   GX    01
SYS$DEQ                            ********   GX    01
SYS$ENQ                            ********    X    01
SYS$SETEF                          ********   GX    01
TPT$L_LOCK                         ********    X    01
TPT$L_QUERY_LCK                    ********    X    01
TPT$L_REC_WAT                      ********    X    01
UNLOCK                             000003FA 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
RMSRMS0                         00000574  ( 1396.)  01 (  1.)    PIC   USR   CON   REL   GBL NOSHR   EXE   RD   NOWRT NOVEC BYTE
$ABS$                           00000000  (      0.)  02 (  2.)  NOPIC   USR   CON   ABS   LCL NOSHR   EXE   RD     WRT NOVEC BYTE

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

Phase                        Page faults    CPU Time       Elapsed Time
-----                        -----------    --------       ------------
Initialization                        29    00:00:00.09    00:00:01.35
Command processing                   107    00:00:00.69    00:00:05.06
Pass 1                               415    00:00:15.67    00:00:54.17
Symbol table sort                      0    00:00:01.99    00:00:04.18
Pass 2                               231    00:00:04.07    00:00:14.01
Symbol table output                   20    00:00:00.19    00:00:00.69
Psect synopsis output                  1    00:00:00.06    00:00:00.36
Cross-reference output                 0    00:00:00.00    00:00:00.00
Assembler run totals                 805    00:00:22.77    00:01:19.83

The working set limit was 1650 pages.
86780 bytes (170 pages) of virtual memory were used to buffer the intermediate code.
There were 70 pages of symbol table space allocated to hold 1337 non-local and 91 local symbols.
1547 source lines were read in Pass 1, producing 18 object records in Pass 2.
36 pages of virtual memory were used to define 35 macros.

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

Macro library name                               Macros defined
------------------                               --------------
_$255$DUA28:[RMS.OBJ]RMS.MLB;1                             17
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                              1
_$255$DUA28:[SYSLIB]STARLET.MLB;2                          13
TOTALS (all libraries)                                     31

1493 GETS were required to define 31 macros.

There were no errors, warnings or information messages.

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

0319 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

RM0RECLCK
LIS

RM0RSET
LIS

RM0SCAN
LIS

RM0PRFLNM
LIS

RM0RCLCK2
LIS

RM0RABCHK
LIS

RM0RELEAS
LIS

RM0NAMSTR
LIS