```
   SSSSSSSSSSSS  YYY         YYY     SSSSSSSSSSSS  LLL                 OOOOOOOOO        AAAAAAAAA
   SSSSSSSSSSSS  YYY         YYY     SSSSSSSSSSSS  LLL                 OOOOOOOOO        AAAAAAAAA
   SSSSSSSSSSSS  YYY         YYY     SSSSSSSSSSSS  LLL                 OOOOOOOOO        AAAAAAAAA
SSS              YYY         YYY  SSS              LLL              OOO         OOO  AAA         AAA
SSS              YYY         YYY  SSS              LLL              OOO         OOO  AAA         AAA
SSS              YYY         YYY  SSS              LLL              OOO         OOO  AAA         AAA
SSS                 YYY   YYY     SSS              LLL              OOO         OOO  AAA         AAA
SSS                 YYY   YYY     SSS              LLL              OOO         OOO  AAA         AAA
SSS                 YYY   YYY     SSS              LLL              OOO         OOO  AAA         AAA
   SSSSSSSSS           YYY           SSSSSSSSS     LLL              OOO         OOO  AAA         AAA
   SSSSSSSSS           YYY           SSSSSSSSS     LLL              OOO         OOO  AAA         AAA
   SSSSSSSSS           YYY           SSSSSSSSS     LLL              OOO         OOO  AAA         AAA
            SSS        YYY                    SSS  LLL              OOO         OOO  AAAAAAAAAAAAAAA
            SSS        YYY                    SSS  LLL              OOO         OOO  AAAAAAAAAAAAAAA
            SSS        YYY                    SSS  LLL              OOO         OOO  AAAAAAAAAAAAAAA
            SSS        YYY                    SSS  LLL              OOO         OOO  AAA         AAA
            SSS        YYY                    SSS  LLL              OOO         OOO  AAA         AAA
            SSS        YYY                    SSS  LLL              OOO         OOO  AAA         AAA
SSSSSSSSSSSS           YYY        SSSSSSSSSSSS     LLLLLLLLLLLLLLL     OOOOOOOOO     AAA         AAA
SSSSSSSSSSSS           YYY        SSSSSSSSSSSS     LLLLLLLLLLLLLLL     OOOOOOOOO     AAA         AAA
SSSSSSSSSSSS           YYY        SSSSSSSSSSSS     LLLLLLLLLLLLLLL     OOOOOOOOO     AAA         AAA
```

_$2
Sym

SS$
SS$

SS$
SS$
SS$
SS$
SS$
SS$
SYS
SYS
SYS
TRY
UNL
WRI

FILEID**CSPCJFMAS

```
;  1          0001  0 MODULE  CSPCJFMAS
;  2          0002  0                 (IDENT= 'V04-000'
;  3          0003  0                 ,LANGUAGE (BLISS32)
;  4          0004  0                 ,ADDRESSING_MODE (EXTERNAL= GENERAL)
;  5          0005  0                 ) =
;  6          0006  1 BEGIN
;  7          0007  1 !
;  8          0008  1 !*****************************************************************************
;  9          0009  1 !*                                                                           *
; 10          0010  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
; 11          0011  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
; 12          0012  1 !*  ALL RIGHTS RESERVED.                                                     *
; 13          0013  1 !*                                                                           *
; 14          0014  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
; 15          0015  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
; 16          0016  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
; 17          0017  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
; 18          0018  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
; 19          0019  1 !*  TRANSFERRED.                                                             *
; 20          0020  1 !*                                                                           *
; 21          0021  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
; 22          0022  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
; 23          0023  1 !*  CORPORATION.                                                             *
; 24          0024  1 !*                                                                           *
; 25          0025  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
; 26          0026  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
; 27          0027  1 !*                                                                           *
; 28          0028  1 !*                                                                           *
; 29          0029  1 !*****************************************************************************
; 30          0030  1 !++
; 31          0031  1 !
; 32          0032  1 ! FACILITY:     Action routine for Common Journal Facility executing in
; 33          0033  1 !               Cluster Server Process
; 34          0034  1 !
; 35          0035  1 ! ABSTRACT:     Remaster journals when one or more nodes have been removed
; 36          0036  1 !               from a VAXcluster
; 37          0037  1 !
; 38          0038  1 ! AUTHOR:       Paul R. Beck
; 39          0039  1 !
; 40          0040  1 ! DATE:          27-JUN-1983            Last Edit:  9-SEP-1983 17:30:12
; 41          0041  1 !
; 42          0042  1 ! REVISION HISTORY:
; 43          0043  1 !
; 44          0044  1 !       V03-004 ADE0001         Alan D. Eldridge        10-Feb-1984
; 45          0045  1 !               Change module name.  Rename CSP$GQ_GRANTQ to CSP$GQ_RESUME.
; 46          0046  1 !
; 47          0047  1 !       V03-003 PRB0250         Paul Beck       9-Sep-1983  18:32
; 48          0048  1 !               Fix various problems relating to levels of .indirection.
; 49          0049  1 !               Correct calling of cluster failover resumption routine.
; 50          0050  1 !
; 51          0051  1 !       V03-002 PRB0xxx Paul Beck                15-AUG-1983 15:20
; 52          0052  1 !               Fix various problems with levels of indirection.
; 53          0053  1 !
; 54          0054  1 !       V03-001 PRB0234 Paul Beck                1-AUG-1983 15:20
; 55          0055  1 !               Fix bug in calculating size of journal name,
; 56          0056  1 !               remove reference to JNLRM$B_JNLTYP.
; 57          0057  1 !
```

```
;   58          0058  1 !--
```

```
;    60          0059  1 %SBTTL  'Local definitions and macros'
;    61          0060  1 !
;    62          0061  1 ! Require files:
;    63          0062  1 !
;    64          0063  1 !LIBRARY
;    65          0064  1 !    'SYS$LIBRARY:LIB';                 ! defire system services (in JNLPREFIX)
;    66          0065  1
;    74          4101  1 SWITCHES LIST (SOURCE);
;    75          4102  1 !
;    76          4103  1 ! Linkages used
;    77          4104  1 !
;    78          4105  1 LINKAGE
;    79          4106  1     JSB_2                = JSB (REGISTER=2),
;    80          4107  1     ALONONPAGED          = JSB (REGISTER=1; REGISTER=2): NOPRESERVE(0,1,3),
;    81          4108  1     DEANONPAGED          = JSB (REGISTER=0): NOPRESERVE(0,1,2,3);
;    82          4109  1 !
;    83          4110  1 ! Declare forward routines
;    84          4111  1 !
;    85          4112  1 FORWARD ROUTINE
;    86          4113  1     REMASTER_CHAIN,                          ! remaster jnls of one type
;    87          4114  1     RESUME_FAILOVER,                         ! issue QIO to resume failover sequence
;    88          4115  1     RESTART_BASE         : NOVALUE,          ! reschedule base thread
;    89          4116  1     TEST_DEVICE,                             ! verify accessibility of jnl home device
;    90          4117  1     REMASTER_JOURNAL,                        ! remaster single journal
;    91          4118  1     REMASTER_CONTROL,                        ! remaster control journal
;    92          4119  1     CALL_RCP             : NOVALUE,          ! send request to RCP for RU journal
;    93          4120  1     RELOCK_CHAIN         : NOVALUE,          ! convert lock on chain to exmode
;    94          4121  1     FREE_CHAIN           : NOVALUE,          ! convert lock on chain to null
;    95          4122  1     MUMBLE;
;    96          4123  1 !
;    97          4124  1 ! Declare external routines
;    98          4125  1 !
;    99          4126  1 EXTERNAL ROUTINE
;   100          4127  1     LIB$GET_VM, LIB$FREE_VM,                 ! RTL allocation routines
;   101          4128  1     LIB$GET_EF, LIB$FREE_EF,                 ! RTL event flag routines
;   102          4129  1     SYS$ENQ, SYS$DEQ,                        ! lock manager system services
;   103          4130  1     EXE$ALONONPAGED      : ALONONPAGED,      ! allocate nonpaged pool
;   104          4131  1     EXE$DEANONPAGED      : DEANONPAGED,      ! deallocated nonpaged pool
;   105          4132  1     CJF$ASSJNL,                              ! assign journal
;   106          4133  1     CJF$DEASJNL,                             ! deassign journal
;   107          4134  1     CJF$ALLJDR,                              ! allocate journal drive
;   108          4135  1     CJF$DEALJDR,                             ! deallocate journal drive
;   109          4136  1     CJF$RECOVERW,                            ! issue recover request to RCP
;   110          4137  1     CJF$LOCK_PROTO,                          ! lock prototype UCB
;   111          4138  1     CJF$CREJNL,                              ! create journal
;   112          4139  1     LOCK_IODB            : NOVALUE,          ! lock I/O database
;   113          4140  1     UNLOCK_IODB          : NOVALUE,          ! unlock I/O database
;   114          4141  1     DELETE_IODB          : NOVALUE,          ! delete UCB-related CJF structures
;   115          4142  1     DELETE_UCB           : NOVALUE,          ! delete CJF UCB.
;   116          4143  1     CREATE_RUDEV,                            ! create device name of RU jnl
;   117          4144  1     CSP$$RESUME,                             ! async AST address within CSP
;   118          4145  1     CSP$$END_CONTEXT,                        ! kill current thread and reschedule
;   119          4146  1     CSP$$WAIT,                               ! asynch wait within CSP
;   120          4147  1     CSP$$FORK,                               ! create new thread
;   121          4148  1     CJF$RESUME_FAILOVER : NOVALUE;           ! resume failover sequence
;   122          4149  1 !
;   123          4150  1 ! External variables.
```

```
;  124          4151  1 !
;  125          4152  1 EXTERNAL
;  126          4153  1     CSP$GQ_RESUME,                          ! scheduler queue
;  127          4154  1     CSP$GL_CURCTX,                          ! current context
;  128          4155  1     CSP$GL_BASE_FP,                         ! frame ptr for scheduler call
;  129          4156  1     RESUME_ADDRESS,                         ! addr at which to resume failover
;  130          4157  1     FAILOVER_ID;                            ! ID of most recent failover sequence
;  131          4158  1 !
;  132          4159  1 ! Own storage accessible to all callers and all routines in this module.
;  133          4160  1 !
;  134          4161  1 OWN
;  135          4162  1     LOCAL_EFN           : BYTE,             ! event flag for this routine
;  136          4163  1     UCBQ                : VECTOR [2,LONG]   ! queue of UCB clones
;  137          4164  1                           INITIAL (UCBQ, UCBQ),
;  138          4165  1     UCBMASTERQ          : VECTOR [2,LONG]   ! queue of locally mastered RU UCBs
;  139          4166  1                           INITIAL (UCBMASTERQ, UCBMASTERQ),
;  140          4167  1     CALLED_COUNT        : INITIAL ( 0 ),    ! # times this routine called
;  141          4168  1     REMASTERING         : BYTE INITIAL ( 0 );   ! flag: failover in progress
;  142          4169  1
;  143          4170  1 LITERAL
;  144          4171  1     !
;  145          4172  1     ! Define size of JSB plus maximal device name descriptor list.
;  146          4173  1     !
;  147          4174  1     JSB_LENGTH          = JSB$C_LENGTH + (JNL$C MAX_COPIES*8) + 4;
;  148          4175  1 !
;  149          4176  1 ! Macro to restart remastering loop at the  op.
;  150          4177  1 !
;  151        M 4178  1 MACRO   RESTART_REMASTER_LOOP =
;  152        M 4179  1     BEGIN
;  153        M 4180  1     LOCAL_CALLED_COUNT = .CALLED_COUNT;
;  154        M 4181  1     LEAVE REMASTER_LOOP;
;  155          4182  1     END; % ;
;  156          4183  1 !
;  157          4184  1 ! Macro to issue call with arguments from kernel mode.
;  158          4185  1 !
;  159        M 4186  1 MACRO   KRNL_CALL (K_ROUTINE) =
;  160        M 4187  1     BEGIN
;  161        M 4188  1     EXTERNAL ROUTINE
;  162        M 4189  1         SYS$CMKRNL    : ADDRESSING_MODE (ABSOLUTE);
;  163        M 4190  1     BUILTIN SP ;
;  164        M 4191  1
;  165        M 4192  1     SYS$CMKRNL (K_ROUTINE , .SP , %LENGTH - 1
;  166        M 4193  1                 %IF %LENGTH GTR 1 %THEN , %REMAINING %FI)
;  167          4194  1     END%;
;  168          4195  1 !
;  169          4196  1 ! ********** temps **********
;  170          4197  1 !
;  171        L 4198  1 %IF NOT %DECLARED (CJF$M_FAILOVER)
;  172        U 4199  1 %THEN
;  173        U 4200  1     EXTERNAL LITERAL CJF$M_FAILOVER;
;  174          4201  1 %FI
;  175        L 4202  1 %IF NOT %DECLARED (IO$M_GETMINFO)
;  176        U 4203  1 %THEN
;  177        U 4204  1     EXTERNAL LITERAL IO$M_GETMINFO;
;  178          4205  1 %FI
```

```
;  180     4206  1 %SBTTL  'CSP$CJFREMASTER - remaster journals in a VAXcluster'
;  181     4207  1 !++
;  182     4208  1 ! CSP$CJFREMASTER
;  183     4209  1 !
;  184     4210  1 ! FUNCTIONAL DESCRIPTION:
;  185     4211  1 !       This routine is an action routine running in the context of the
;  186     4212  1 !       Cluster Server Process (CSP). It is invoked from the Journal Driver
;  187     4213  1 !       when JNLDRIVER is notified that a cluster failover is in progress
;  188     4214  1 !       ("cluster failover" is the removal of one or more nodes from a running
;  189     4215  1 !       VAXcluster). Note that unlike many CSP action routines, this routine is
;  190     4216  1 !       invoked in the same node as the driver; it runs in the CSP in order to
;  191     4217  1 !       provide a process context in which to remaster journals. The JNLACP is
;  192     4218  1 !       not used to provide this process context so that the remastering code
;  193     4219  1 !       can conveniently call normal CJF services (which result in JNLACP
;  194     4220  1 !       activity) without making the JNLACP multithreaded.
;  195     4221  1 !
;  196     4222  1 !       Journals may need to be remastered following a node failure in a cluster
;  197     4223  1 !       because that node may have been master to one or more cluster-wide
;  198     4224  1 !       journals in use from other nodes. If the journal file is accessible from
;  199     4225  1 !       another node, that node may remaster the journal.
;  200     4226  1 !
;  201     4227  1 !       The function of the routine is to try to become the master of any
;  202     4228  1 !       journals which are currently unmastered and available. All nodes in the
;  203     4229  1 !       cluster attempt to do this; the lock manager is used to keep them honest.
;  204     4230  1 !       The order of events is:
;  205     4231  1 !               0.      Note that when this routine is entered, the VAXcluster
;  206     4232  1 !                       in the middle of the failover festivities. Locks from
;  207     4233  1 !                       the failed node(s) have been deleted, and most locking
;  208     4234  1 !                       activity is permitted; however, "protected" locks
;  209     4235  1 !                       (LCK$V_PROTECT) are still being stalled to allow the RCP
;  210     4236  1 !                       to reestablish locks from Recovery Units in the failed
;  211     4237  1 !                       node(s). (Ref steps 1-3)
;  212     4238  1 !               1.      Obtain the journal master lock (CJF$MASTER) to lock out
;  213     4239  1 !                       any journal startup code on nodes just starting up.
;  214     4240  1 !               2.      Remaster Recovery Unit Journals. This involves obtaining
;  215     4241  1 !                       a lock on the first unmastered RU journal we can,
;  216     4242  1 !                       remastering it (if it's accessible), releasing the lock,
;  217     4243  1 !                       and trying for the next. The loop continues until all
;  218     4244  1 !                       unmastered RU Journals are either mastered or marked
;  219     4245  1 !                       "unavailable".
;  220     4246  1 !               3.      Invoke the Recovery Control Process once for each
;  221     4247  1 !                       Recovery Unit Journal mastered locally. The RCP will
;  222     4248  1 !                       reestablish locks for those Recovery Units not marked
;  223     4249  1 !                       completed from nodes not in the cluster.
;  224     4250  1 !               4.      "Resume" the Failover table.
;  225     4251  1 !               5.      Remaster Control Journals so that journal devices are
;  226     4252  1 !                       reallocated as necessary.
;  227     4253  1 !               6.      Remaster AI, BI, and AT Journals.
;  228     4254  1 !               7.      Release the Journaling Master Lock.
;  229     4255  1 ! CALLING SEQUENCE:
;  230     4256  1 !       MOVAL   CSD,R2
;  231     4257  1 !       JSB     CSP$CJFREMASTER
;  232     4258  1 ! FORMAL PARAMETERS:
;  233     4259  1 !       P1      = address of CSD structure identifying the calling parameters.
;  234     4260  1 !                 No parameters are required; just being invoked is sufficient.
;  235     4261  1 ! COMPLETION CODES:
;  236     4262  1 !
```

```
;  237          4263  1 !--
;  238          4264  1 GLOBAL ROUTINE CSP$CJFREMASTER ( CSD: REF BLOCK [,BYTE] ): JSB_2 =
;  239          4265  2 BEGIN
;  240          4266  2 !
;  241          4267  2 ! Local storage
;  242          4268  2 !
;  243          4269  2 LOCAL
;  244          4270  2     STATUS,                                   ! completion status
;  245          4271  2     RU_STATUS,                                ! ...
;  246          4272  2     CL_STATUS,                                ! ...
;  247          4273  2     AI_STATUS,                                ! ...
;  248          4274  2     BI_STATUS,                                ! ...
;  249          4275  2     AT_STATUS,                                ! ...
;  250          4276  2     LOCAL_CALLED_COUNT  : LONG,               ! impure copy of CALLED_COUNT
;  251          4277  2     HOME_CONTEXT        : REF BLOCK [,BYTE],  ! address of context block for this thread
;  252          4278  2     MASTER_LKSB         : VECTOR [2,LONG];    ! lock status block
;  253          4279  2
;  254          4280  2 LABEL
;  255          4281  2     REMASTER_LOOP;
;  256          4282  2
;  257          4283  2 HOME_CONTEXT = .CSP$GL_CURCTX;
;  258          4284  2 CALLED_COUNT = .CALLED_COUNT + 1;
;  259          4285  2 !
;  260          4286  2 ! Always save the lastest failover ID and sequence resume address. These
;  261          4287  2 ! reside with the routine CJF$RESUME_FAILOVER. If a new failover sequence has
;  262          4288  2 ! started we don't want to resume an old one... These values are filled in by
;  263          4289  2 ! the journal driver when it is first called from the failover sequence.
;  264          4290  2 !
;  265          4291  2 RESUME_ADDRESS = .CSD [CSD$L_P1];
;  266          4292  2 FAILOVER_ID = .CSD [CSD$L_P2];
;  267          4293  2 !
;  268          4294  2 ! Bail out if somebody is currently in the throes of remastering journals.
;  269          4295  2 !
;  270          4296  2 IF NOT .REMASTERING THEN REMASTERING = 1 ELSE RETURN SS$_NORMAL;
;  271          4297  2 LOCAL_CALLED_COUNT = .CALLED_COUNT;
;  272          4298  2 !
;  273          4299  2 ! Get an event flag for synchronous waits in kernel mode.
;  274          4300  2 !
;  275          4301  2 LIB$GET_EF (LOCAL_EFN);
;  276          4302  2 !
;  277          4303  2 ! Get the CJF master lock (from Kernel mode)
;  278          4304  2 !
;  279        P 4305  2 STATUS = KRNL_CALL ( SYS$ENQ,
;  280        P 4306  2                      0,                       ! EFN
;  281        P 4307  2                      LCK$K_CWMODE,            ! LKMODE = concurrent write lock
;  282        P 4308  2                      HOME_CONTEXT [CLX$Q_IOSB], ! LKSB = lock status block
;  283        P 4309  2                      LCK$M_SYSTEM,            ! FLAGS  = system-wide lock
;  284        P 4310  2                      %ASCID 'CJF$MASTER',     ! RESNAM = name of lock
;  285        P 4311  2                      0,                       ! PARID  = no parent
;  286        P 4312  2                      CSP$$RESUME,             ! ASTADR = completion AST in CSP
;  287        P 4313  2                      .HOME_CONTEXT,           ! ASTPRM = allows CSP to resume us
;  288        P 4314  2                      0,                       ! BLKAST = no blocking AST
;  289        P 4315  2                      PSL$C_KERNEL,            ! ACMODE = kernel mode lock
;  290          4316  2                      0);                      ! reserved parameter
;  291          4317  2 IF .STATUS THEN CSP$$WAIT();
;  292          4318  2 STATUS = .HOME_CONTEXT [CLX$W_IOSB_STAT];
;  293          4319  2 IF NOT .STATUS THEN MUMBLE();                 ! why can't we get it?
```

```
;  294          4320  2 !
;  295          4321  2 !                          *** TOP OF LOOP ***
;  296          4322  2 !
;  297          4323  2 ! We do the entire process in a loop, making sure that if any journals become
;  298          4324  2 ! unmastered while we're here (i.e. a second failover), we catch them.
;  299          4325  2 ! This implies a couple of caveats:
;  300          4326  2 !       1.      We "get out" when we've made a pass wherein NO unmastered
;  301          4327  2 !               journals not marked "unavailable" were seen.
;  302          4328  2 !       2.      This "no unmastered journals" final pass must be made such
;  303          4329  2 !               that NO CSP WAITS ARE DONE (fully synchronous). Otherwise we
;  304          4330  2 !               might miss something (and a new call to CSP$CJFREMASTER would
;  305          4331  2 !               just exit because we have REMASTERING set).
;  306          4332  2 !       3.      Because of the special nature of the handling of RU journals,
;  307          4333  2 !               we look for unmastered RU journals at a higher rate than
;  308          4334  2 !               the "peon" journals (AI/BI/AT/CL).
;  309          4335  2 WHILE 1
;  310          4336  2 DO
;  311          4337  2
;  312          4338  2 REMASTER_LOOP:
;  313          4339  2
;  314          4340  3     BEGIN
;  315          4341  3     !
;  316          4342  3     ! Remaster Recovery Unit journals and call RCP for each locally mastered RUJ.
;  317          4343  3     !
;  318          4344  3     RU_STATUS = 0;
;  319          4345  3     UNTIL .RU_STATUS EQL SS$_NOSUCHDEV
;  320          4346  3     DO
;  321          4347  3         RU_STATUS = KRNL_CALL ( REMASTER_CHAIN,  DT$_RUJNL );
;  322          4348  3     !
;  323          4349  3     ! Resume the failover table. This tells the Cluster connection manager to
;  324          4350  3     ! continue executing those steps required to complete the failover. In particular,
;  325          4351  3     ! full locking will once more be permitted.
;  326          4352  3     !
;  327          4353  3     KRNL_CALL ( CJF$RESUME_FAILOVER );
;  328          4354  3     !
;  329          4355  3     ! Remaster control journals and reallocate journal devices.
;  330          4356  3     !
;  331          4357  3     CL_STATUS = 0;
;  332          4358  3     UNTIL .CL_STATUS EQL SS$_NOSUCHDEV
;  333          4359  3     DO
;  334          4360  3         CL_STATUS = KRNL_CALL ( REMASTER_CHAIN,  DT$_CLJNL );
;  335          4361  3     !
;  336          4362  3     ! If any new requests have been seen, restart from the top immediately.
;  337          4363  3     !
;  338          4364  3     IF .CALLED_COUNT NEQ .LOCAL_CALLED_COUNT THEN RESTART_REMASTER_LOOP;
;  339          4365  3     !
;  340          4366  3     ! Remaster AI journals.
;  341          4367  3     !
;  342          4368  3     AI_STATUS = KRNL_CALL ( REMASTER_CHAIN,  DT$_AIJNL );
;  343          4369  3     !
;  344          4370  3     ! If any new requests have been seen, restart from the top immediately.
;  345          4371  3     !
;  346          4372  3     IF .CALLED_COUNT NEQ .LOCAL_CALLED_COUNT THEN RESTART_REMASTER_LOOP;
;  347          4373  3     !
;  348          4374  3     ! Remaster BI journals.
;  349          4375  3     !
;  350          4376  3     BI_STATUS = KRNL_CALL ( REMASTER_CHAIN,  DT$_BIJNL );
```

```
;  351          4377  3      !
;  352          4378  3      ! If any new requests have been seen, restart from the top immediately.
;  353          4379  3      !
;  354          4380  3      IF .CALLED_COUNT NEQ .LOCAL_CALLED_COUNT THEN RESTART_REMASTER_LOOP;
;  355          4381  3      !
;  356          4382  3      ! Remaster AT journals.
;  357          4383  3      !
;  358          4384  3      AT_STATUS = KRNL_CALL ( REMASTER_CHAIN,  DT$_ATJNL );
;  359          4385  3      !
;  360          4386  3      ! If any new requests have been seen, restart from the top immediately.
;  361          4387  3      !
;  362          4388  3      IF .CALLED_COUNT NEQ .LOCAL_CALLED_COUNT THEN RESTART_REMASTER_LOOP;
;  363          4389  3      !
;  364          4390  3      ! See if we did anything this pass. If not, we're done.
;  365          4391  3      !
;  366          4392  4      IF (.RU_STATUS EQL SS$_NOSUCHDEV)                    ! no RUJs this pass
;  367          4393  4          AND (.CL_STATUS EQL SS$_NOSUCHDEV)               ! no control jnls this pass
;  368          4394  4          AND (.AI_STATUS EQL SS$_NOSUCHDEV)               ! no AIJs this pass
;  369          4395  4          AND (.BI_STATUS EQL SS$_NOSUCHDEV)               ! no BIJs this pass
;  370          4396  4          AND (.AT_STATUS EQL SS$_NOSUCHDEV)               ! no ATJs this pass
;  371          4397  3      THEN
;  372          4398  3          !
;  373          4399  3          ! No attempts were made to remaster any journals this pass.
;  374          4400  3          ! Ergo, we're done.
;  375          4401  3          !
;  376          4402  3          EXITLOOP;
;  377          4403  2      END;                                             ! WHILE 1
;  378          4404  2  !
;  379          4405  2  ! Reset the mutex.
;  380          4406  2  !
;  381          4407  2  LIB$FREE_EF (LOCAL_EFN);
;  382          4408  2  REMASTERING = 0;
;  383          4409  2  !
;  384          4410  2  ! Release the lock
;  385          4411  2  !
;  386          4412  3      BEGIN
;  387          4413  3      BIND
;  388          4414  3          LOCK_ID = HOME_CONTEXT [CLX$Q_IOSB] + 4;
;  389          4415  3      KRNL_CALL ( SYS$DEQ, .LOCK_ID, 0, PSL$C_KERNEL, 0 );
;  390          4416  2      END;
;  391          4417  2  !
;  392          4418  2  ! That's it.
;  393          4419  2  !
;  394          4420  2  RETURN SS$_NORMAL;
;  395          4421  1  END;


                                                                .TITLE  CSPCJFMAS
                                                                .IDENT  \V04-000\

                                                                .PSECT  CJF$PLIT,NOWRT,NOEXE,  SHR,  PIC,2

                00  00  52  45  54  53  41  4D  24  46  4A  43  00000 P.AAB:  .ASCII  \CJF$MASTER\<0><0>                    ;
                                                    010E000A  0000C P.AAA:  .LONG   17694730                            ;
                                                    00000000'  00010        .ADDRESS P.AAB                              ;

                                                                .PSECT  CJF$OWN,NOEXE,  PIC,2
```

```
                                00000 LOCAL_EFN:
                                      .BLKB   1
                                00001 
                                      .BLKB   3
                    00000000' 00000000' 00004 UCBQ:    .ADDRESS UCBQ, UCBQ                                    ;
                    00000000' 00000000' 0000C UCBMASTERQ:
                                      .ADDRESS UCBMASTERQ, UCBMASTERQ                                        ;
                              00000000  00014 CALLED_COUNT:
                                      .LONG   0                                                              ;
                                    00  00018 REMASTERING:
                                      .BYTE   0                                                              ;

                                      .EXTRN  IOC$CVT_DEVNAM, EXE$ALOP1IMAG
                                      .EXTRN  EXE$DEAP1, LIB$GET_VM
                                      .EXTRN  LIB$FREE_VM, LIB$GET_EF
                                      .EXTRN  LIB$FREE_EF, SYS$ENQ
                                      .EXTRN  SYS$DEQ, EXE$ALONONPAGED
                                      .EXTRN  EXE$DEANONPAGED
                                      .EXTRN  CJF$ASSJNL, CJF$DEASJNL
                                      .EXTRN  CJF$ALLJDR, CJF$DEALJDR
                                      .EXTRN  CJF$RECOVERW, CJF$LOCK_PROTO
                                      .EXTRN  CJF$CREJNL, LOCK_IODB
                                      .EXTRN  UNLOCK_IODB, DELETE_IODB
                                      .EXTRN  DELETE_UCB, CREATE_RUDEV
                                      .EXTRN  CSP$$RESUME, CSP$$END_CONTEXT
                                      .EXTRN  CSP$$WAIT, CSP$$FORK
                                      .EXTRN  CJF$RESUME_FAILOVER
                                      .EXTRN  CSP$GQ_RESUME, CSP$GL_CURCTX
                                      .EXTRN  CSP$GL_BASE_FP, RESUME_ADDRESS
                                      .EXTRN  FAILOVER_ID, SYS$CMKRNL

                                      .PSECT  CJF$CODE,NOWRT,  SHR,  PIC,2

                          01FC    8F  BB 00000 CSP$CJFREMASTER::
                                              PUSHR   #^M<R2,R3,R4,R5,R6,R7,R8>                              ; 4264
                       5E                 08  C2 00004                SUBL2   #8, SP                          ;
                       54 00000000G       00  D0 00007                MOVL    CSP$GL_CURCTX, HOME_CONTEXT     ; 4283
                          00000000'       EF  D6 0000E                 INCL    CALLED_COUNT                    ; 4284
00000000G  00             52               A2  D0 00014                MOVL    82(CSD), RESUME_ADDRESS         ; 4291
00000000G  00             56               A2  D0 0001C                MOVL    86(CSD), FAILOVER_ID            ; 4292
                       03 00000000'       EF  E9 00024                BLBC    REMASTERING, 1$                 ; 4296
                                     0176  31  00029                BRW     11$                             ;
00000000'  EF                          01  90 0002E 1$:              MOVB    #1, REMASTERING                 ;
                       52 00000000'       EF  D0 00035                MOVL    CALLED_COUNT, LOCAL_CALLED_COUNT ; 4297
                          00000000'       EF  9F 0003C                PUSHAB  LOCAL_EFN                       ; 4301
00000000G  00                          01  FB 00042                CALLS   #1, LIB$GET_EF                  ;
                                       7E  7C 00049                CLRQ    -(SP)                           ; 4316
                                       7E  D4 0004B                CLRL    -(SP)                           ;
                                       54  DD 0004D                PUSHL   HOME_CONTEXT                    ;
                             00000000G     00  9F 0004F                PUSHAB  CSP$$RESUME                     ;
                                       7E  D4 00055                CLRL    -(SP)                           ;
                             00000000'     EF  9F 00057                PUSHAB  P.AAA                           ;
                                       10  DD 0005D                PUSHL   #16                             ;
                                30     A4  9F 0005F                PUSHAB  48(HOME_CONTEXT)                ;
                                       02  DD 00062                PUSHL   #2                              ;
                       7E                 0B  7D 00064                MOVQ    #11, -(SP)                     ;
                                       5E  DD 00067                PUSHL   SP                              ;
```

```
                        00000000G  00  9F 00069          PUSHAB  SYS$ENQ
                00000000G  9F      0E  FB 0006F          CALLS   #14, @#SYS$CMKRNL
                           53      50  D0 00076          MOVL    R0, STATUS
                           07      53  E9 00079          BLBC    STATUS, 2$                      ;4317
                00000000G  00      00  FB 0007C          CALLS   #0, CSP$$WAIT
                           53  30  A4  3C 00083 2$:       MOVZWL  48(HOME_CONTEXT), STATUS        ;4318
                           05      53  E8 00087          BLBS    STATUS, 3$                      ;4319
                     0000V CF      00  FB 0008A          CALLS   #0, MUMBLE
                           55      D4  0008F 3$:         CLRL    RU_STATUS                       ;4344
                00000908  8F      55  D1 00091 4$:       CMPL    RU_STATUS, #2312                ;4345
                                  16  13 00098          BEQL    5$
                                  01  DD 0009A          PUSHL   #1                              ;4347
                                  01  DD 0009C          PUSHL   #1
                                  5E  DD 0009E          PUSHL   SP
                           0000V  CF  9F 000A0          PUSHAB  REMASTER_CHAIN
                00000000G  9F      04  FB 000A4          CALLS   #4, @#SYS$CMKRNL
                           55      50  D0 000AB          MOVL    R0, RU_STATUS
                                  E1  11 000AE          BRB     4$
                                  7E  D4 000B0 5$:       CLRL    -(SP)                           ;4353
                                  5E  DD 000B2          PUSHL   SP
                        00000000G  00  9F 000B4          PUSHAB  CJF$RESUME_FAILOVER
                00000000G  9F      03  FB 000BA          CALLS   #3, @#SYS$CMKRNL
                                  53  D4 000C1          CLRL    CL_STATUS                       ;4357
                00000908  8F      53  D1 000C3 6$:       CMPL    CL_STATUS, #2312                ;4358
                                  16  13 000CA          BEQL    7$
                                  05  DD 000CC          PUSHL   #5                              ;4360
                                  01  DD 000CE          PUSHL   #1
                                  5E  DD 000D0          PUSHL   SP
                           0000V  CF  9F 000D2          PUSHAB  REMASTER_CHAIN
                00000000G  9F      04  FB 000D6          CALLS   #4, @#SYS$CMKRNL
                           53      50  D0 000DD          MOVL    R0, CL_STATUS
                                  E1  11 000E0          BRB     6$
                           52 00000000'  EF  D1 000E2 7$:       CMPL    CALLED_COUNT, LOCAL_CALLED_COUNT  ;4364
                                  57  12 000E9          BNEQ    8$
                                  03  DD 000EB          PUSHL   #3                              ;4368
                                  01  DD 000ED          PUSHL   #1
                                  5E  DD 000EF          PUSHL   SP
                           0000V  CF  9F 000F1          PUSHAB  REMASTER_CHAIN
                00000000G  9F      04  FB 000F5          CALLS   #4, @#SYS$CMKRNL
                           58      50  D0 000FC          MOVL    R0, AI_STATUS
                           52 00000000'  EF  D1 000FF          CMPL    CALLED_COUNT, LOCAL_CALLED_COUNT  ;4372
                                  3A  12 00106          BNEQ    8$
                                  02  DD 00108          PUSHL   #2                              ;4376
                                  01  DD 0010A          PUSHL   #1
                                  5E  DD 0010C          PUSHL   SP
                           0000V  CF  9F 0010E          PUSHAB  REMASTER_CHAIN
                00000000G  9F      04  FB 00112          CALLS   #4, @#SYS$CMKRNL
                           57      50  D0 00119          MOVL    R0, BI_STATUS
                           52 00000000'  EF  D1 0011C          CMPL    CALLED_COUNT, LOCAL_CALLED_COUNT  ;4380
                                  1D  12 00123          BNEQ    8$
                                  04  DD 00125          PUSHL   #4                              ;4384
                                  01  DD 00127          PUSHL   #1
                                  5E  DD 00129          PUSHL   SP
                           0000V  CF  9F 0012B          PUSHAB  REMASTER_CHAIN
                00000000G  9F      04  FB 0012F          CALLS   #4, @#SYS$CMKRNL
                           56      50  D0 00136          MOVL    R0, AT_STATUS
                           52 00000000'  EF  D1 00139          CMPL    CALLED_COUNT, LOCAL_CALLED_COUNT  ;4388
```

```
                                    0A    13 00140          BEQL    10$                                     ;
                  52 00000000'      EF    D0 00142 8$:      MOVL    CALLED_COUNT, LOCAL_CALLED_COUNT        ;
                                  FF43    31 00149 9$:      BRW     3$                                      ;
         00000908   8F              55    D1 0014C 10$:     CMPL    RU_STATUS, #2312                        ; 4392
                                    F4    12 00153          BNEQ    9$                                      ;
         00000908   8F              53    D1 00155          CMPL    CL_STATUS, #2312                        ; 4393
                                    EB    12 0015C          BNEQ    9$                                      ;
         00000908   8F              58    D1 0015E          CMPL    AI_STATUS, #2312                        ; 4394
                                    E2    12 00165          BNEQ    9$                                      ;
         00000908   8F              57    D1 00167          CMPL    BI_STATUS, #2312                        ; 4395
                                    D9    12 0016E          BNEQ    9$                                      ;
         00000908   8F              56    D1 00170          CMPL    AT_STATUS, #2312                        ; 4396
                                    D0    12 00177          BNEQ    9$                                      ;
                     00000000'      EF    9F 00179          PUSHAB  LOCAL_EFN                               ; 4407
         00000000G  00              01    FB 0017F          CALLS   #1, LIB$FREE_EF                         ;
                     00000000'      EF    94 00186          CLRB    REMASTERING                             ; 4408
                                    7E    7C 0018C          CLRQ    -(SP)                                   ; 4415
                                    7E    D4 0018E          CLRL    -(SP)                                   ;
                              34    A4    DD 00190          PUSHL   52(HOME_CONTEXT)                        ;
                                    04    DD 00193          PUSHL   #4                                      ;
                                    5E    DD 00195          PUSHL   SP                                      ;
                     00000000G      00    9F 00197          PUSHAB  SYS$DEQ                                 ;
         00000000G  9F              07    FB 0019D          CALLS   #7, @#SYS$CMKRNL                        ;
                    50              01    D0 001A4 11$:     MOVL    #1, R0                                  ; 4420
                    5E              08    C0 001A7          ADDL2   #8, SP                                  ; 4421
                            01FC    8F    BA 001AA          POPR    #^M<R2,R3,R4,R5,R6,R7,R8>               ;
                                          05 001AE          RSB                                             ;

; Routine Size:  431 bytes,    Routine Base:  CJF$CODE + 0000
```

```
;  397          4422  1 %SBTTL  'REMASTER_CHAIN - remaster journals of one type'
;  398          4423  1 !++
;  399          4424  1 ! REMASTER_CHAIN
;  400          4425  1 !
;  401          4426  1 ! FUNCTIONAL DESCRIPTION:
;  402          4427  1 !       Keep trying to remaster journals of the specified type
;  403          4428  1 !       until all journals of this type are either remastered by someone
;  404          4429  1 !       or are known to be unavailable. We operate primarily off a copy of
;  405          4430  1 !       the UCB chain to keep the I/O data base unlocked and know that
;  406          4431  1 !       we're safe since while we're remastering, no new journals (and hence
;  407          4432  1 !       no new UCBs) can show up.
;  408          4433  1 ! CALLING SEQUENCE:
;  409          4434  1 !       STATUS  = KRNL_CALL REMASTER_CHAIN ( JNLTYP );
;  410          4435  1 ! FORMAL PARAMETERS:
;  411          4436  1 !       JNLTYP  = value of journal type to be remastered.
;  412          4437  1 ! COMPLETION CODES:
;  413          4438  1 !       SS$_NORMAL      = one or more journal was touched (an attempt was made,
;  414          4439  1 !                         whether successful or not, to remaster it)
;  415          4440  1 !       SS$_NOSUCHDEV   = no available, unmastered journals were found
;  416          4441  1 !--
;  417          4442  1 ROUTINE REMASTER_CHAIN ( JNLTYP ) =
;  418          4443  2 BEGIN
;  419          4444  2 LOCAL
;  420          4445  2     STATUS,                                   ! completion status
;  421          4446  2     PROTO_UCB           : REF BLOCK [,BYTE],  ! addr of UCB0 for this chain
;  422          4447  2     UCB                 : REF BLOCK [,BYTE],  ! addr of individual UCB
;  423          4448  2     LOC_UCB             : REF BLOCK [,BYTE],  ! addr of copied UCB
;  424          4449  2     LKSB                : VECTOR [2,LONG],    ! lock status block for proto lock
;  425          4450  2     JNL_LKSB            : VECTOR [2,LONG],    ! lock status block for journal lock
;  426          4451  2     JNL_LOCKNAME        : BLOCK [8,BYTE]      ! descriptor of lock name
;  427          4452  2                           PRESET ( [DSC$W_LENGTH]  = 0,
;  428          4453  2                                    [DSC$B_DTYPE]   = 0,
;  429          4454  2                                    [DSC$B_CLASS]   = 0,
;  430          4455  2                                    [DSC$A_POINTER] = 0),
;  431          4456  2     DELETE_LOCAL_UCB    : BYTE INITIAL (0),   ! flag
;  432          4457  2     UNAVAILABLE         : BYTE,               ! flag
;  433          4458  2     SOMETHING_DONE      : BYTE INITIAL (0),   ! flag: suitable journal found
;  434          4459  2     NEXT_UCB            : LONG,               ! pointer to next UCB
;  435          4460  2     ALLOC_SIZE          : LONG;               ! allocation size
;  436          4461  2
;  437          4462  2 BIND
;  438          4463  2     PROTO_LOCK_ID       = LKSB [ 1 ];         ! lock ID for UCB chain lock
;  439          4464  2 LABEL
;  440          4465  2     PROCESS_UCB;
;  441          4466  2 BUILTIN
;  442          4467  2     INSQUE,REMQUE;
;  443          4468  2 !
;  444          4469  2 ! First, lock the I/O data base and this UCB chain
;  445          4470  2 !
;  446          4471  2 STATUS = CJF$LOCK_PROTO ( .JNLTYP,            ! define which chain to lock
;  447          4472  2                           LCK$K_EXMODE,       ! we want an exclusive lock
;  448          4473  2                           PROTO_UCB,          ! return the addr of the base UCB
;  449          4474  2                           PROTO_LOCK_ID );    ! return the lock ID
;  450          4475  2 IF NOT .STATUS THEN MUMBLE();
;  451          4476  2 UNLOCK_IODB();
;  452          4477  2 !
;  453          4478  2 ! Make a copy of each unmastered UCB in this chain while we have everything
```

```
;  454          4479  2 ! locked up. Keep a backpointer to the real UCB in UCB$L_LINK.
;  455          4480  2 !
;  456          4481  2 UCB = .PROTO_UCB;
;  457          4482  2 UNTIL (UCB = .UCB [UCB$L_LINK]) EQL 0
;  458          4483  2 DO
;  459          4484  3     BEGIN
;  460          4485  3     BIND
;  461          4486  3         UCB_DEVCHAR = UCB [UCB$L_DEVCHAR]: BLOCK [4,BYTE]; ! "available" flag in this longword
;  462          4487  3
;  463          4488  4     IF (.UCB [UCB$V_JNL_UNMAST]) AND (.UCB_DEVCHAR [DEV$V_AVL])
;  464          4489  3     THEN
;  465          4490  4         BEGIN
;  466          4491  4         !
;  467          4492  4         ! This UCB is unmastered and not marked unavailable. Copy it.
;  468          4493  4         !
;  469          4494  4         SOMETHING_DONE = 1;
;  470          4495  4         EXE$ALONONPAGED ( UCB$C_JNL_LENGTH; LOC_UCB );
;  471          4496  4         CH$MOVE ( UCB$C_JNL_LENGTH, .UCB, .LOC_UCB );
;  472          4497  4         LOC_UCB [UCB$L_LINK] = .UCB;
;  473          4498  4         INSQUE ( .LOC_UCB, UCBQ );
;  474          4499  3         END;
;  475          4500  2     END;
;  476          4501  2 !
;  477          4502  2 ! Rather than releasing the lock on the proto UCB, we downgrade it to a
;  478          4503  2 ! null lock so we can use it as the parent lock for Journal Locks.
;  479          4504  2 !
;  480          4505  2 FREE_CHAIN ( LKSB );
;  481          4506  2 !
;  482          4507  2 ! Start an "endless loop" which is only exited when we recognize that all
;  483          4508  2 ! journals in this UCB chain have either been remastered (by us or somebody
;  484          4509  2 ! else), or marked unavailable.
;  485          4510  2 !
;  486          4511  2 LOC_UCB = UCBQ;                          ! init "this UCB" pointer
;  487          4512  2 WHILE 1
;  488          4513  2 DO
;  489          4514  3     BEGIN
;  490          4515  3     !
;  491          4516  3     ! For each journal in he list, attempt to remaster.
;  492          4517  3     ! First, we must get the lock for the journal. If we can't get the lock,
;  493          4518  3     ! we move on to the next journal in the list. When we run out of journals,
;  494          4519  3     ! we try again! Eventually (it sez here) all journals will have been taken
;  495          4520  3     ! care of.
;  496          4521  3     !
;  497          4522  3     ! If the queue is empty, we're done (we've run out of journals).
;  498          4523  3     !
;  499          4524  3     IF .UCBQ EQL UCBQ THEN EXITLOOP;
;  500          4525  3     IF .LOC_UCB EQL UCBQ THEN LOC_UCB = ..LOC_UCB;
;  501          4526  3
;  502          4527  3 PROCESS_UCB:
;  503          4528  3
;  504          4529  4         BEGIN
;  505          4530  4         !
;  506          4531  4         ! The journal lock consists of the journal name (as found in the
;  507          4532  4         ! UCB), zero-filled as needed, with the UCB chain lock as its parent.
;  508          4533  4         !
;  509          4534  4         LOCAL
;  510          4535  4             NAME_SIZE   : BYTE;                     ! length of journal name
```

```
;  511          4536  4
;  512          4537  4          NAME_SIZE = .LOC_UCB [UCB$T_JNL_NAM];
;  513          4538  4          IF (.NAME_SIZE EQL 0) THEN NAME_SIZE = UCB$S_JNL_NAM;
;  514          4539  4          DELETE_LOCAL_UCB = 0;
;  515          4540  4          UNAVAILABLE = 0;
;  516          4541  4          NEXT_UCB = ..LOC_UCB;
;  517          4542  4          CH$FILL ( 0, UCB$S_JNL_NAM - .NAME_SIZE, LOC_UCB [UCB$B_JNL_NAM] + .NAME_SIZE );
;  518          4543  4          JNL_LOCKNAME [DSC$W_LENGTH] = UCB$S_JNL_NAM;
;  519          4544  4          JNL_LOCKNAME [DSC$A_POINTER] = LOC_UCB [UCB$T_JNL_NAM];
;  520        P 4545  4          STATUS = $ENQ ( LKMODE  = LCK$K_PWMODE,          ! protected write lock
;  521        P 4546  4                          LKSB    = JNL_LKSB,           ! lock status block
;  522        P 4547  4                          FLAGS   = LCK$M_NOQUEUE OR LCK$M_SYSTEM, ! now or not at all
;  523        P 4548  4                          RESNAM  = JNL_LOCKNAME,       ! name of lock
;  524          4549  4                          PARID   = .PROTO_LOCK_ID );   ! parent = proto
;  525          4550  4          IF NOT .STATUS THEN LEAVE PROCESS_UCB WITH (DELETE_LOCAL_UCB = 0);
;  526          4551  4
;  527          4552  4          ! We got the lock.
;  528          4553  4          !
;  529          4554  4          ! Now, try to remaster the journal. First, however, double-check
;  530          4555  4          ! the real UCB to see if someone else mastered it first. (As long
;  531          4556  4          ! as we have the journal lock, they won't.)
;  532          4557  4          !
;  533          4558  4          RELOCK_CHAIN ( LKSB );
;  534          4559  4          UCB = .LOC_UCB [UCB$L_LINK];
;  535          4560  4          STATUS = .UCB [UCB$V_JNL_UNMAST];
;  536          4561  4          !
;  537          4562  4          ! We have the information we want (in STATUS): release the chain.
;  538          4563  4          !
;  539          4564  4          FREE_CHAIN ( LKSB );
;  540          4565  4          !
;  541          4566  4          ! If it's no longer unmastered, go on to the next UCB
;  542          4567  4          !
;  543          4568  4          IF NOT .STATUS THEN LEAVE PROCESS_UCB WITH (DELETE_LOCAL_UCB = 1);
;  544          4569  4          !
;  545          4570  4          ! It's still unmastered. Here, we try to remaster it.
;  546          4571  4          !
;  547          4572  4          SELECTONE .JNLTYP OF
;  548          4573  4              SET
;  549          4574  4              [DT$_AIJNL, DT$_BIJNL, DT$_ATJNL, DT$_RUJNL]:
;  550          4575  4                  IF NOT REMASTER_JOURNAL ( .LOC_UCB, LKSB )
;  551          4576  4                  THEN
;  552          4577  4                      LEAVE PROCESS_UCB WITH (UNAVAILABLE = 1);
;  553          4578  4              [DT$_CLJNL]:
;  554          4579  5                  BEGIN
;  555          4580  5                  REMASTER_CONTROL ( .LOC_UCB, LKSB );
;  556          4581  5                  DELETE_LOCAL_UCB = 1;
;  557          4582  4                  END;
;  558          4583  4              TES;
;  559          4584  3          END;                                    ! PROCESS_UCB
;  560          4585  3      !
;  561          4586  3      ! Release the journal lock
;  562          4587  3      !
;  563          4588  3      $DEQ ( LKID = .JNL_LKSB [ 1 ] );
;  564          4589  3      !
;  565          4590  3      ! Mark journal unavailable if requested
;  566          4591  3      !
;  567          4592  3      UCB = .LOC_UCB [UCB$L_LINK];
```

```
;  568          4593  3     IF .UNAVAILABLE
;  569          4594  3     THEN
;  570          4595  4         BEGIN
;  571          4596  4         BIND
;  572          4597  4             UCB_DEVCHAR = UCB [UCB$L_DEVCHAR]: BLOCK [4,BYTE];
;  573          4598  4         !
;  574          4599  4         ! First, get the lock exclusively again.
;  575          4600  4         !
;  576          4601  4         RELOCK_CHAIN ( LKSB );
;  577          4602  4         !
;  578          4603  4         ! Mark the UCB unavailable.
;  579          4604  4         !
;  580          4605  4         UCB_DEVCHAR [DEV$V_AVL] = 0;
;  581          4606  4         !
;  582          4607  4         ! Release chain lock
;  583          4608  4         !
;  584          4609  4         FREE_CHAIN ( LKSB );
;  585          4610  4         DELETE_LOCAL_UCB = 1;
;  586          4611  3         END;
;  587          4612  3     !
;  588          4613  3     ! Get rid of the UCB if needed.
;  589          4614  3     !
;  590          4615  3     IF .DELETE_LOCAL_UCB
;  591          4616  3     THEN
;  592          4617  4         BEGIN
;  593          4618  4         REMQUE ( .LOC_UCB, LOC_UCB );
;  594          4619  4         EXE$DEANONPAGED ( .LOC_UCB );
;  595          4620  3         END;
;  596          4621  3     !
;  597          4622  3     ! Get pointer to next local UCB and continue.
;  598          4623  3     !
;  599          4624  3     LOC_UCB = .NEXT_UCB;
;  600          4625  2     END;                                        ! While 1
;  601          4626  2 !
;  602          4627  2 ! Call the RCP for each journal mastered by this node.
;  603          4628  2 !
;  604          4629  2 IF .JNLTYP EQL DT$_RUJNL
;  605          4630  2 THEN
;  606          4631  3     BEGIN
;  607          4632  3     LOC_UCB = UCBMASTERQ;
;  608          4633  3     UNTIL (LOC_UCB = .LOC_UCB) EQL UCBMASTERQ
;  609          4634  3     DO
;  610          4635  3         CALL_RCP ( .LOC_UCB );
;  611          4636  2     END;
;  612          4637  2 !
;  613          4638  2 ! Release our local allocations.
;  614          4639  2 !
;  615          4640  2 UNTIL REMQUE ( .UCBQ, UCB ) DO EXE$DEANONPAGED ( .UCB );
;  616          4641  2 UNTIL REMQUE ( .UCBMASTERQ, UCB ) DO EXE$DEANONPAGED ( .UCB );
;  617          4642  2 !
;  618          4643  2 ! Release the chain lock for real.
;  619          4644  2 !
;  620          4645  2 $DEQ ( LKID = .LKSB [ 1 ] );
;  621          4646  2 !
;  622          4647  2 ! Done
;  623          4648  2 !
;  624          4649  2 RETURN (IF .SOMETHING_DONE THEN SS$_NORMAL ELSE SS$_NOSUCHDEV);
```

```
:  625          4650  1 END;


                                                 0FFC 00000 REMASTER_CHAIN:
                                                            .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11          : 4442
                                 5E                  20  C2 00002          SUBL2   #32, SP                       :
                                           08        AE  7C 00005          CLRQ    JNL_LOCKNAME                  : 4455
                                                     58  94 00008          CLRB    DELETE_LOCAL_UCB              :
                                                     5A  94 0000A          CLRB    SOMETHING_DONE                :
                                           1C        AE  9F 0000C          PUSHAB  PROTO_LOCK_ID                 : 4471
                                           08        AE  9F 0000F          PUSHAB  PROTO_UCB                     :
                                                     05  DD 00012          PUSHL   #5                            :
                                           04        AC  DD 00014          PUSHL   JNLTYP                        :
                              00000000G    00           04  FB 00017          CALLS   #4, CJF$LOCK_PROTO            :
                                           59           50  D0 0001E          MOVL    R0, STATUS                    :
                                           05           59  E8 00021          BLBS    STATUS, 1$                    : 4475
                                    0000V  CF           00  FB 00024          CALLS   #0, MUMBLE                    :
                              00000000G    00           00  FB 00029 1$:      CALLS   #0, UNLOCK_IODB               : 4476
                                           57     04    AE  D0 00030          MOVL    PROTO_UCB, UCB                : 4481
                                           57     30    A7  D0 00034 2$:      MOVL    48(UCB), UCB                  : 4482
                                                        2E  13 00038          BEQL    3$                            :
                                 F5    69    A7           01  E1 0003A          BBC     #1, 105(UCB), 2$              : 4488
                                 F0    3A    A7           02  E1 0003F          BBC     #2, 58(UCB), 2$               :
                                           5A           01  90 00044          MOVB    #1, SOMETHING_DONE            : 4494
                                           51     011C  8F  3C 00047          MOVZWL  #284, R1                      : 4495
                                        00000000G    00  16 0004C          JSB     EXE$ALONONPAGED               :
                                           56           52  D0 00052          MOVL    R2, R6                        :
                                 66              67     011C  8F  28 00055          MOVC3   #284, (UCB), (LOC_UCB)        : 4496
                                       30        A6           57  D0 0005B          MOVL    UCB, 48(LOC_UCB)              : 4497
                                 00000000'       EF           66  0E 0005F          INSQUE  (LOC_UCB), @UCBQ              : 4498
                                                              CC  11 00066          BRB     2$                            : 4482
                                                        18    AE  9F 00068 3$:      PUSHAB  LKSB                          : 4505
                                          0000V  CF           01  FB 0006B          CALLS   #1, FREE_CHAIN                :
                                           56 00000000'       EF  9E 00070          MOVAB   UCBQ, LOC_UCB                 : 4511
                                           50 00000000'       EF  9E 00077 4$:      MOVAB   UCBQ, R0                      : 4524
                                           50 00000000'       EF  D1 0007E          CMPL    UCBQ, R0                      :
                                                              03  12 00085          BNEQ    5$                            :
                                                           0OED  31 00087          BRW     14$                           :
                                           50 00000000'       EF  9E 0008A 5$:      MOVAB   UCBQ, R0                      : 4525
                                           50                 56  D1 00091          CMPL    LOC_UCB, R0                   :
                                                              03  12 00094          BNEQ    6$                            :
                                           56                 66  D0 00096          MOVL    (LOC_UCB), LOC_UCB            :
                                           50        00B8     C6  90 00099 6$:      MOVB    184(LOC_UCB), NAME_SIZE       : 4537
                                                              03  12 0009E          BNEQ    7$                            : 4538
                                           50                 12  90 000A0          MOVB    #18, NAME_SIZE                :
                                                              58  94 000A3 7$:      CLRB    DELETE_LOCAL_UCB              : 4539
                                                              5B  94 000A5          CLRB    UNAVAILABLE                   : 4540
                                           6E                 66  D0 000A7          MOVL    (LOC_UCB), NEXT_UCB           : 4541
                                           50                 50  9A 000AA          MOVZBL  NAME_SIZE, R0                 : 4542
                                 51        12                 50  C3 000AD          SUBL3   R0, #18, R1                   :
                              51 00        6E                 00  2C 000B1          MOVC5   #0, (SP), #0, R1, 185(R0)[LOC_UCB]  :
                                                        00B9 C046                 000B6                                 :
                                       08  AE                 12  B0 000BA          MOVW    #18, JNL_LOCKNAME             : 4543
                                       0C  AE           00B8  C6  9E 000BE          MOVAB   184(LOC_UCB), JNL_LOCKNAME+4  : 4544
```

```
                                                 7E  7C 000C4          CLRQ    -(SP)                                          ; 4549
                                                 7E  7C 000C6          CLRQ    -(SP)                                          ;
                                                 7E  D4 000C8          CLRL    -(SP)                                          ;
                                              30 AE  DD 000CA          PUSHL   PROTO_LOCK_ID                                  ;
                                              20 AE  9F 000CD          PUSHAB  JNL_LOCKNAME                                   ;
                                                 14  DD 000D0          PUSHL   #20                                            ;
                                              30 AE  9F 000D2          PUSHAB  JNL_LKSB                                       ;
                                                 04  DD 000D5          PUSHL   #4                                             ;
                                                 7E  D4 000D7          CLRL    -(SP)                                          ;
                                  00000000G   00 0B  FB 000D9          CALLS   #11, SYS$ENQ                                   ;
                                              59 50  D0 000E0          MOVL    R0, STATUS                                     ;
                                              04 59  E8 000E3          BLBS    STATUS, 8$                                     ; 4550
                                                 58  94 000E6          CLRB    DELETE_LOCAL_UCB                               ;
                                                 4C  11 000E8          BRB     11$                                            ;
                                              18 AE  9F 000EA 8$:      PUSHAB  LKSB                                           ; 4558
                                    0000V     CF 01  FB 000ED          CALLS   #1, RELOCK_CHAIN                               ;
                                              57 30  A6 D0 000F2       MOVL    48(LOC_UCB), UCB                               ; 4559
                                59        69  A7 01  01 EF 000F6       EXTZV   #1, #1, 105(UCB), STATUS                       ; 4560
                                              18 AE  9F 000FC          PUSHAB  LKSB                                           ; 4564
                                    0000V     CF 01  FB 000FF          CALLS   #1, FREE_CHAIN                                 ;
                                              2C 59  E9 00104          BLBC    STATUS, 10$                                    ; 4568
                                              50 04  AC D0 00107       MOVL    JNLTYP, R0                                     ; 4572
                                                 17  15 0010B          BLEQ    9$                                             ; 4574
                                              04 50  D1 0010D          CMPL    R0, #4                                         ;
                                                 12  14 00110          BGTR    9$                                             ;
                                              18 AE  9F 00112          PUSHAB  LKSB                                           ; 4575
                                                 56  DD 00115          PUSHL   LOC_UCB                                        ;
                                    0000V     CF 02  FB 00117          CALLS   #2, REMASTER_JOURNAL                           ;
                                              17 50  E8 0011C          BLBS    R0, 11$                                        ;
                                              5B 01  90 0011F          MOVB    #1, UNAVAILABLE                                ; 4577
                                                 12  11 00122          BRB     11$                                            ;
                                              05 50  D1 00124 9$:      CMPL    R0, #5                                         ; 4578
                                                 0D  12 00127          BNEQ    11$                                            ;
                                                 18  AE 9F 00129       PUSHAB  LKSB                                           ; 4580
                                                 56  DD 0012C          PUSHL   LOC_UCB                                        ;
                                    0000V     CF 02  FB 0012E          CALLS   #2, REMASTER_CONTROL                           ;
                                              58 01  90 00133 10$:     MOVB    #1, DELETE_LOCAL_UCB                           ; 4581
                                                 7E  7C 00136 11$:     CLRQ    -(SP)                                          ; 4588
                                                 7E  D4 00138          CLRL    -(SP)                                          ;
                                                 20  AE DD 0013A       PUSHL   JNL_LKSB+4                                     ;
                                  00000000G   00 04  FB 0013D          CALLS   #4, SYS$DEQ                                    ;
                                              57 30  A6 D0 00144       MOVL    48(LOC_UCB), UCB                               ; 4592
                                              17 5B  E9 00148          BLBC    UNAVAILABLE, 12$                               ; 4593
                                                 18  AE 9F 0014B       PUSHAB  LKSB                                           ; 4601
                                    0000V     CF 01  FB 0014E          CALLS   #1, RELOCK_CHAIN                               ;
                                      3A      A7 04  8A 00153          BICB2   #4, 58(UCB)                                    ; 4605
                                                 18  AE 9F 00157       PUSHAB  LKSB                                           ; 4609
                                    0000V     CF 01  FB 0015A          CALLS   #1, FREE_CHAIN                                 ;
                                              58 01  90 0015F          MOVB    #1, DELETE_LOCAL_UCB                           ; 4610
                                              0C 58  E9 00162 12$:     BLBC    DELETE_LOCAL_UCB, 13$                          ; 4615
                                              56 66  0F 00165          REMQUE  (LOC_UCB), LOC_UCB                             ; 4618
                                              50 56  D0 00168          MOVL    LOC_UCB, R0                                    ; 4619
                                         00000000G   00 16 0016B       JSB     EXE$DEANONPAGED                                ;
                                              56 6E  D0 00171 13$:     MOVL    NEXT_UCB, LOC_UCB                              ; 4624
                                                 FF00 31 00174         BRW     4$                                             ; 4512
                                              01 04  AC D1 00177 14$:  CMPL    JNLTYP, #1                                     ; 4629
                                                 1C  12 0017B          BNEQ    16$                                            ;
```

```
                                56 00000000'   EF   9E 0017D          MOVAB   UCBMASTERQ, LOC_UCB                    ; 4632
                                50 00000000'   EF   9E 00184 15$:     MOVAB   UCBMASTERQ, R0                      ; 4633
                                50             56   D1 0018B          CMPL    LOC_UCB, R0                         ;
                                               09   13 0018E          BEQL    16$                                 ;
                                               56   DD 00190          PUSHL   LOC_UCB                             ; 4635
                          0000V CF             01   FB 00192          CALLS   #1, CALL_RCP                        ;
                                               EB   11 00197          BRB     15$                                 ;
                                57 00000000'   FF   0F 00199 16$:     REMQUE  @UCBQ, UCB                          ; 4640
                                               0B   1D 001A0          BVS     17$                                 ;
                                50             57   D0 001A2          MOVL    UCB, R0                             ;
                                   00000000G   00   16 001A5          JSB     EXE$DEANONPAGED                     ;
                                               EC   11 001AB          BRB     16$                                 ;
                                57 00000000'   FF   0F 001AD 17$:     REMQUE  @UCBMASTERQ, UCB                    ; 4641
                                               0B   1D 001B4          BVS     18$                                 ;
                                50             57   D0 001B6          MOVL    UCB, R0                             ;
                                   00000000G   00   16 001B9          JSB     EXE$DEANONPAGED                     ;
                                               EC   11 001BF          BRB     17$                                 ;
                                               7E   7C 001C1 18$:     CLRQ    -(SP)                               ; 4645
                                               7E   D4 001C3          CLRL    -(SP)                               ;
                                      28       AE   DD 001C5          PUSHL   LKSB+4                              ;
                    00000000G     00           04   FB 001C8          CALLS   #4, SYS$DEQ                         ;
                                  04           5A   E9 001CF          BLBC    SOMETHING_DONE, 19$                 ; 4649
                                50             01   D0 001D2          MOVL    #1, R0                              ;
                                               04   001D5            RET                                         ;
                                50      0908   8F   3C 001D6 19$:     MOVZWL  #2312, R0                           ;
                                               04   001DB            RET                                         ; 4650

; Routine Size:  476 bytes,    Routine Base:  CJF$CODE + 01AF
```

```
;  627          4651  1 %SBTTL  'REMASTER_JOURNAL - try to remaster AI, BI, AT, RU journal'
;  628          4652  1 !++
;  629          4653  1 ! REMASTER_JOURNAL
;  630          4654  1 !
;  631          4655  1 ! FUNCTIONAL DESCRIPTION:
;  632          4656  1 !       Try to remaster this journal with the following steps:
;  633          4657  1 !       1.  Issue request to the driver to collect information about this
;  634          4658  1 !           journal from other nodes in the cluster
;  635          4659  1 !       2.  Test accessibility of the device on which the journal file resides.
;  636          4660  1 !       3.  Create a JSB structure from the information in the Remaster Block
;  637          4661  1 !           associated with this journal's UCB
;  638          4662  1 !       4.  Issue a Create Journal service to effect the remastering operation.
;  639          4663  1 !       5.  If successful, move the local copy of the UCB to the queue of
;  640          4664  1 !           locally mastered journals. This is only done if we're dealing with
;  641          4665  1 !           a Recovery Unit journal.
;  642          4666  1 !       THIS ROUTINE RUNS IN KERNEL MODE.
;  643          4667  1 ! CALLING SEQUENCE:
;  644          4668  1 !       STATUS  = REMASTER_JOURNAL ( LOC_UCB, LKSB );
;  645          4669  1 ! FORMAL PARAMETERS:
;  646          4670  1 !       LOC_UCB = address of local copy of journal's UCB.
;  647          4671  1 !       LKSB    = address of lock status block for the chain lock
;  648          4672  1 ! COMPLETION CODES:
;  649          4673  1 !       0       = We didn't remaster this journal
;  650          4674  1 !       1       = We did remaster this journal
;  651          4675  1 !--
;  652          4676  1 ROUTINE REMASTER_JOURNAL ( LOC_UCB: REF BLOCK [,BYTE], LKSB: VECTOR [2,LONG] ) =
;  653          4677  2 BEGIN
;  654          4678  2
;  655          4679  2 LITERAL
;  656          4680  2     DEVNAM_LENGTH       = 15;
;  657          4681  2
;  658          4682  2 LOCAL
;  659          4683  2     JSB                 : BLOCK [JSB$C_LENGTH,BYTE], ! internal JSB
;  660          4684  2     CHANNEL             : WORD,                      ! channel to journal device
;  661          4685  2     LOCAL_IOSB          : VECTOR [2,LONG],           ! I/O status block
;  662          4686  2     DDB                 : REF BLOCK [,BYTE],         ! device data block
;  663          4687  2     UCB                 : REF BLOCK [,BYTE],         ! actual UCB for this journal
;  664          4688  2     FAO_BUFFER          : BLOCK [DEVNAM_LENGTH,BYTE], ! build 'CJxuuuu' here
;  665          4689  2     JNL_DEVNAM          : BLOCK [8,BYTE]
;  666          4690  2                           PRESET ( [DSC$W_LENGTH]  = DEVNAM_LENGTH,
;  667          4691  2                                    [DSC$B_DTYPE]   = 0,
;  668          4692  2                                    [DSC$B_CLASS]   = 0,
;  669          4693  2                                    [DSC$A_POINTER] = FAO_BUFFER),
;  670          4694  2     JNLRM               : REF BLOCK [,BYTE],         ! remaster block
;  671          4695  2     DEVNAM              : REF BLOCK [,BYTE],         ! pointer into remaster block
;  672          4696  2     DEVNAM_LIST         : BLOCK [((CJF$C_MAXCOPIES+1)*8),BYTE], ! device name descriptor list
;  673          4697  2     DEVNAM_DESC         : REF BLOCK [,BYTE],         ! pointer into DEVNAM_LIST
;  674          4698  2     STATUS              : LONG;
;  675          4699  2 !
;  676          4700  2 ! First get a channel to the journal. We don't need (or want) a "journal
;  677          4701  2 ! channel" using $ASSJNL, since that assumes a mastered journal. So we
;  678          4702  2 ! must first create the name of the journal device from the structures
;  679          4703  2 ! we already have at hand.
;  680          4704  2 !
;  681          4705  2 ! *** NOTE *** We will access the I/O database to read the DDB, which
;  682          4706  2 ! shouldn't either move or change, so we're not locking it.
;  683          4707  2 !
```

```
;  684          4708  2 DDB = .LOC_UCB [UCB$L_DDB];
;  685          4709  2 !
;  686          4710  2 ! Create the device name. Note that JNL_DEVNAM [DSC$W_LENGTH] gets
;  687          4711  2 ! re-initialized every time the subroutine is entered, so we can change
;  688          4712  2 ! it at will here (by using it with "outlen").
;  689          4713  2 !
;  690        P 4714  2 $FAO (  %ASCID '!AC!UW',                          ! ctrstr
;  691        P 4715  2         JNL_DEVNAM [DSC$W_LENGTH],                ! outlen
;  692        P 4716  2         JNL_DEVNAM,                               ! outbuf
;  693        P 4717  2         DDB [DDB$T_NAME],                         ! p1
;  694          4718  2         .LOC_UCB [UCB$W_UNIT] );                  ! p2
;  695          4719  2 !
;  696          4720  2 ! Get the channel.
;  697          4721  2 !
;  698          4722  2 IF NOT $ASSIGN ( CHAN = CHANNEL, DEVNAM = JNL_DEVNAM ) THEN RETURN 0;
;  699          4723  2 !
;  700          4724  2 ! Ask the driver to collect information from other
;  701          4725  2 ! nodes in the cluster so that it can become master
;  702          4726  2 ! of the journal.
;  703          4727  2 !
;  704        P 4728  2 STATUS = $QIOW( CHAN    = .CHANNEL,
;  705        P 4729  2                 FUNC    = IO$_SNDJNLMSG OR IO$M_GETMINFO,
;  706        P 4730  2                 IOSB    = LOCAL_IOSB,
;  707          4731  2                 EFN     = .LOCAL_EFN );
;  708          4732  2 STATUS = .LOCAL_IOSB;
;  709          4733  2 $DASSGN ( CHAN = .CHANNEL );
;  710          4734  2 !
;  711          4735  2 ! See if it worked.
;  712          4736  2 !
;  713          4737  2 IF NOT .STATUS THEN RETURN 0;
;  714          4738  2 !
;  715          4739  2 ! We have accumulated all the data from other nodes needed to be master of
;  716          4740  2 ! this journal with the exception of write buffers etc. Next, see if we have
;  717          4741  2 ! access to the device on which the journal file exists.
;  718          4742  2 !
;  719          4743  2 IF NOT TEST_DEVICE ( .LOC_UCB ) THEN RETURN 0;
;  720          4744  2 !
;  721          4745  2 ! We have access to the device. Now create the journal with this node as its
;  722          4746  2 ! master. To do this, we first create a JSB from the information in the JNLRM
;  723          4747  2 ! structure.
;  724          4748  2 !
;  725          4749  2 ! Start out with a zeroed JSB
;  726          4750  2 !
;  727          4751  2 CH$FILL ( 0, JSB_LENGTH, JSB );
;  728          4752  2 !
;  729          4753  2 ! Fill in what we can; the rest defaults.
;  730          4754  2 !
;  731          4755  2 JNLRM = .LOC_UCB [UCB$L_JNL_RMBLK];
;  732          4756  2 JSB [JSB$W_JNLNAMLEN] = .(LOC_UCB [UCB$T_JNL_NAM])<0,8>;
;  733          4757  2 JSB [JSB$L_JNLNAM] = LOC_UCB [UCB$B_JNL_NAM];
;  734          4758  2 JSB [JSB$B_JNLTYP] = .LOC_UCB [UCB$B_DEVTYPE];
;  735          4759  2 JSB [JSB$B_JNLDEV] = (IF .JNLRM [JNLRM$V_TAPJNL] THEN JSB$C_TAPE ELSE JSB$C_DISK);
;  736          4760  2 JSB [JSB$L_MASK] = .LOC_UCB [UCB$L_JNL_MASK];
;  737          4761  2 JSB [JSB$V_TMPFIL] = .JNLRM [JNLRM$V_TMPFIL];
;  738          4762  2 JSB [JSB$V_DIFACP] = .JNLRM [JNLRM$V_DIFACP];
;  739          4763  2 JSB [JSB$W_ACPNAMLEN] = .JNLRM [JNLRM$W_ACPNAMLEN];
;  740          4764  2 JSB [JSB$L_ACPNAM] = JNLRM [JNLRM$W_ACPNAMOFF] + .JNLRM [JNLRM$W_ACPNAMOFF];
```

```
;  741          4765  2 JSB [JSB$B_COPIES] = .JNLRM [JNLRM$B_COPIES];
;  742          4766  2 JSB [JSB$L_PRINAMDES] = DEVNAM_LIST;
;  743          4767  2 JSB [JSB$V_REMASTER] = 1;
;  744          4768  2 !
;  745          4769  2 ! Zero the descriptor list (including a terminating zero).
;  746          4770  2 !
;  747          4771  2 CH$FILL ( 0, (CJF$C_MAXCOPIES*8)+4, DEVNAM_LIST );
;  748          4772  2 IF .JNLRM [JNLRM$V_DSKJNL]
;  749          4773  2 THEN
;  750          4774  2     INCR INDEX FROM 0 TO .JNLRM [JNLRM$B_COPIES] - 1
;  751          4775  2     DO
;  752          4776  3         BEGIN
;  753          4777  3         !
;  754          4778  3         ! For disk-based journals, we need to create the descriptor list
;  755          4779  3         ! of disk device names.
;  756          4780  3         !
;  757          4781  3         DEVNAM = .JNLRM + JNLRM$K_DSKJNLLST + (.INDEX*JNLRM$K_DSKENTLEN);
;  758          4782  3         DEVNAM_DESC = DEVNAM_LIST + (.INDEX*8);
;  759          4783  3         DEVNAM_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  760          4784  3         DEVNAM_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  761          4785  3         DEVNAM_DESC [DSC$W_LENGTH] = .DEVNAM [JNLRM$W_DEVNAMLEN];
;  762          4786  3         DEVNAM_DESC [DSC$A_POINTER] = .DEVNAM [JNLRM$W_DEVNAMOFF] + DEVNAM [JNLRM$W_DEVNAMOFF];
;  763          4787  3
;  764          4788  3         ! Do anything with JNLRM$W_FILVEROFF/LEN ?
;  765          4789  3         !
;  766          4790  3         MUMBLE();
;  767          4791  3
;  768          4792  2         END;
;  769          4793  2 IF .JNLRM [JNLRM$V_TAPJNL]
;  770          4794  2 THEN
;  771          4795  3     BEGIN
;  772          4796  3     !
;  773          4797  3     ! For tape-based journals, we need to supply the name of the tape
;  774          4798  3     ! journal group.
;  775          4799  3     !
;  776          4800  3     DEVNAM_LIST [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  777          4801  3     DEVNAM_LIST [DSC$B_CLASS] = DSC$K_CLASS_S;
;  778          4802  3     DEVNAM_LIST [DSC$W_LENGTH] = .JNLRM [JNLRM$W_TAPGRPLEN];
;  779          4803  3     DEVNAM_LIST [DSC$A_POINTER] = .JNLRM [JNLRM$W_TAPGRPOFF] + JNLRM [JNLRM$W_TAPGRPOFF];
;  780          4804  2     END;
;  781          4805  2 !
;  782          4806  2 ! JSB is hereby created
;  783          4807  2 ! Now create the journal with this node as its master.
;  784          4808  2 !
;  785          4809  2 STATUS = CJF$CREJNL ( CHANNEL, JSB );
;  786          4810  2 IF NOT .STATUS THEN RETURN 0;
;  787          4811  2 !
;  788          4812  2 ! Successfully remastered lock. Move the UCB copy to the master list.
;  789          4813  2 !
;  790          4814  2 REMQUE ( .LOC_UCB, LOC_UCB );
;  791          4815  2 INSQUE ( .LOC_UCB, UCB$MASTERQ );
;  792          4816  2 !
;  793          4817  2 ! Deallocate the remaster block, if any.
;  794          4818  2 !
;  795          4819  2 RELOCK_CHAIN ( .LKSB );
;  796          4820  2 UCB = .LOC_UCB [UCB$L_LINK];
;  797          4821  2 UCB [UCB$L_JNL_RMBLK] = 0;
```

```
;  798          4822  2 EXE$DEANONPAGED ( .JNLRM );
;  799          4823  2 FREE_CHAIN ( .LKSB );
;  800          4824  2 RETURN 1;
;  801          4825  1 END;


                                                        .PSECT  CJF$PLIT,NOWRT,NOEXE,  SHR,  PIC,2

                    00  00  57  55  21  43  41  21  00014 P.AAD:  .ASCII  \!AC!UW\<0><0>                                    ;
                                        010E0006  0001C P.AAC:  .LONG   17694726                                          ;
                                        00000000'  00020        .ADDRESS P.AAD                                            ;

                                                        .EXTRN  SYS$FAO, SYS$ASSIGN
                                                        .EXTRN  SYS$QIOW, SYS$DASSGN

                                                        .PSECT  CJF$CODE,NOWRT,  SHR,  PIC,2

                                                OFFC 00000 REMASTER_JOURNAL:
                                                        .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11                      ; 4676
                          5E      FF7C    CE  9E 00002          MOVAB   -132(SP), SP                                      ;
                       14 AE              0F  D0 00007          MOVL    #15, JNL_DEVNAM                                   ; 4693
                       18 AE        1C    AE  9E 0000B          MOVAB   FAO_BUFFER, JNL_DEVNAM+4                          ;
                          57        04    AC  D0 00010          MOVL    LOC_UCB, R7                                       ; 4708
                          50        28    A7  D0 00014          MOVL    40(R7), DDB                                       ;
                          7E        54    A7  3C 00018          MOVZWL  84(R7), -(SP)                                     ; 4718
                                    14    A0  9F 0001C          PUSHAB  20(DDB)                                           ;
                                    1C    AE  9F 0001F          PUSHAB  JNL_DEVNAM                                        ;
                                    20    AE  9F 00022          PUSHAB  JNL_DEVNAM                                        ;
                                00000000'   EF  9F 00025          PUSHAB  P.AAC                                             ;
                    00000000G   00          05  FB 0002B          CALLS   #5, SYS$FAO                                       ;
                                            7E  7C 00032          CLRQ    -(SP)                                             ; 4722
                                    08      AE  9F 00034          PUSHAB  CHANNEL                                           ;
                                    20      AE  9F 00037          PUSHAB  JNL_DEVNAM                                        ;
                          00000000G   00          04  FB 0003A          CALLS   #4, SYS$ASSIGN                                    ;
                                      36        50  E9 00041          BLBC    R0, 1$                                            ;
                                                7E  7C 00044          CLRQ    -(SP)                                             ; 4731
                                                7E  7C 00046          CLRQ    -(SP)                                             ;
                                                7E  7C 00048          CLRQ    -(SP)                                             ;
                                                7E  7C 0004A          CLRQ    -(SP)                                             ;
                                            4C  AE  9F 0004C          PUSHAB  LOCAL_IOSB                                        ;
                                      7E  1003  8F  3C 0004F          MOVZWL  #4099, -(SP)                                      ;
                                      7E    28  AE  3C 00054          MOVZWL  CHANNEL, -(SP)                                    ;
                                      7E  00000000'   EF  9A 00058          MOVZBL  LOCAL_EFN, -(SP)                                  ;
                          00000000G   00          0C  FB 0005F          CALLS   #12, SYS$QIOW                                     ;
                                      58          50  D0 00066          MOVL    R0, STATUS                                        ;
                                      58    2C    AE  D0 00069          MOVL    LOCAL_IOSB, STATUS                                ; 4732
                                      7E          6E  3C 0006D          MOVZWL  CHANNEL, -(SP)                                    ; 4733
                          00000000G   00          01  FB 00070          CALLS   #1, SYS$DASSGN                                    ;
                                      03          58  E8 00077          BLBS    STATUS, 2$                                        ; 4737
                                                0112  31 0007A 1$:       BRW     9$                                                ;
                                                57  DD 0007D 2$:       PUSHL   R7                                                ; 4743
                                0000V   CF          01  FB 0007F          CALLS   #1, TEST_DEVICE                                   ;
                                      F3          50  E9 00084          BLBC    R0, 1$                                            ;
005C    8F          00                6E          00  2C 00087          MOVC5   #0, (SP), #0, #92, JSB                            ; 4751
                                                34  AE     0008E                                                                  ;
                                      56    0104  C7  D0 00090          MOVL    260(R7), JNLRM                                    ; 4755
```

```
                                                34   AE    00B8   C7  9B 00095          MOVZBW  184(R7), JSB                                ;4756
                                                38   AE    00B9   C7  9E 0009B          MOVAB   185(R7), JSB+4                              ;4757
                                                3E   AE      41   A7  90 000A1          MOVB    65(R7), JSB+10                              ;4758
                                      05        0C   A6           01  E1 000A6          BBC     #1, 12(JNLRM), 3$                           ;4759
                                                     50           02  D0 000AB          MOVL    #2, R0                                      ;
                                                                  03  11 000AE          BRB     4$                                          ;
                                                     50           01  D0 000B0 3$:      MOVL    #1, R0                                      ;
                                                3F   AE           50  90 000B3 4$:      MOVB    R0, JSB+11                                  ;
                                                40   AE    00D4   C7  D0 000B7          MOVL    212(R7), JSB+12                             ;4760
                                50        0C   A6           01  EF 000BD          EXTZV   #2, #1, 12(JNLRM), R0                       ;4761
                                60   AE        01           04  F0 000C3          INSV    R0, #4, #1, JSB+44                          ;
                                50        0C   A6           01  EF 000C9          EXTZV   #3, #1, 12(JNLRM), R0                       ;4762
                                60   AE        01           06  F0 000CF          INSV    R0, #6, #1, JSB+44                          ;
                                                64   AE      16   A6  B0 000D5          MOVW    22(JNLRM), JSB+48                           ;4763
                                                50   AE      14   A6  3C 000DA          MOVZWL  20(JNLRM), R0                               ;4764
                                                68   AE      14 A046  9E 000DE          MOVAB   20(R0)[JNLRM], JSB+52                       ;
                                                6E   AE      0E   A6  90 000E4          MOVB    14(JNLRM), JSB+58                           ;4765
                                                78   AE      04   AE  9E 000E9          MOVAB   DEVNAM_LIST, JSB+68                         ;4766
                                                61   AE           04  88 000EE          BISB2   #4, JSB+45                                  ;4767
                                0C                   00           6E  2C 000F2          MOVC5   #0, (SP), #0, #12, DEVNAM_LIST              ;4771
                                                                  04  AE    000F7          ;
                                                                  2E  0C   A6  E9 000F9          BLBC    12(JNLRM), 7$                       ;4772
                                                                  55  0E   A6  9A 000FD          MOVZBL  14(JNLRM), R5                       ;4774
                                                                  53          01  CE 00101          MNEGL   #1, INDEX                           ;
                                                                              21  11 00104          BRB     6$                                  ;
                                                                  54      18 A643  7E 00106 5$:      MOVAQ   24(JNLRM)[INDEX], DEVNAM            ;4781
                                                                  52      04 AE43  7E 001CB          MOVAQ   DEVNAM_LIST[INDEX], DEVNAM_DESC     ;4782
                                                02   A2      010E   8F  B0 00110          MOVW    #270, 2(DEVNAM_DESC)                ;4783
                                                                  62      02   A4  B0 00116          MOVW    2(DEVNAM), (DEVNAM_DESC)            ;4785
                                                                  50          64  3C 0011A          MOVZWL  (DEVNAM), R0                        ;4786
                                      04   A2                   50          54  C1 0011D          ADDL3   DEVNAM, R0, 4(DEVNAM_DESC)          ;
                                           0000V                CF          00  FB 00122          CALLS   #0, MUMBLE                          ;4790
                                DB                              53          55  F2 00127 6$:      AOBLSS  R5, INDEX, 5$                       ;4774
                                17        0C   A6           01  E1 0012B 7$:      BBC     #1, 12(JNLRM), 8$                           ;4793
                                          06   AE      010E   8F  B0 00130          MOVW    #270, DEVNAM_LIST+2                         ;4800
                                          04   AE        1A   A6  B0 00136          MOVW    26(JNLRM), DEVNAM_LIST                      ;4802
                                                50        18   A6  9E 0013B          MOVAB   24(JNLRM), R0                               ;4803
                                                51             60  3C 0013F          MOVZWL  (R0), R1                                    ;
                                08   AE           51          50  C1 00142          ADDL3   R0, R1, DEVNAM_LIST+4                       ;
                                                          34   AE  9F 00147 8$:      PUSHAB  JSB                                         ;4809
                                                          04   AE  9F 0014A          PUSHAB  CHANNEL                                     ;
                                     00000000G            00      02  FB 0014D          CALLS   #2, CJF$CREJNL                              ;
                                                          58          50  D0 00154          MOVL    R0, STATUS                          ;
                                                          35          58  E9 00157          BLBC    STATUS, 9$                          ;4810
                                                    04    AC          67  0F 0015A          REMQUE  (R7), LOC_UCB                       ;4814
                                                00000000'   EF    04   BC  0E 0015E          INSQUE  @LOC_UCB, UCBMASTERQ                ;4815
                                                          08   AC  DD 00166          PUSHL   LKSB                                        ;4819
                                                     0000V   CF          01  FB 00169          CALLS   #1, RELOCK_CHAIN                    ;
                                                          50        04   AC  D0 0016E          MOVL    LOC_UCB, R0                         ;4820
                                                          50        30   A0  D0 00172          MOVL    48(R0), UCB                         ;
                                                               0104   C0  D4 00176          CLRL    260(UCB)                            ;4821
                                                          50             56  D0 0017A          MOVL    JNLRM, R0                           ;4822
                                                               00000000G 00  16 0017D          JSB     EXE$DEANONPAGED                     ;
                                                          08   AC  DD 00183          PUSHL   LKSB                                        ;4823
                                                     0000V   CF          01  FB 00186          CALLS   #1, FREE_CHAIN                      ;
                                                          50             01  D0 0018B          MOVL    #1, R0                              ;4824
                                                                         04  0018E          RET                                         ;
```

```
                                50  D4 0018F 9$:    CLRL    R0                      ; 4825
                                    04 00191        RET                             ;

; Routine Size:  402 bytes,    Routine Base:  CJF$CODE + 038B
```

```
;  803     4826  1 %SBTTL  'REMASTER_CONTROL - remaster control journal/reallocate/mount for jnling'
;  804     4827  1 !++
;  805     4828  1 ! REMASTER_CONTROL
;  806     4829  1 !
;  807     4830  1 ! FUNCTIONAL DESCRIPTION:
;  808     4831  1 !       For a particular control journal, walk through the list of allocated
;  809     4832  1 !       devices and perform the necessary allocate/mount functions to make
;  810     4833  1 !       them available for journaling.
;  811     4834  1 !       THIS ROUTINE RUNS IN KERNEL MODE.
;  812     4835  1 ! CALLING SEQUENCE:
;  813     4836  1 !       STATUS = REMASTER_CONTROL ( LOC_UCB, LKSB );
;  814     4837  1 ! FORMAL PARAMETERS:
;  815     4838  1 !       LOC_UCB = address of local copy of this control journal's UCB
;  816     4839  1 !       LKSB    = address of lock status block for the chain lock
;  817     4840  1 ! COMPLETION CODES:
;  818     4841  1 !       1       = success
;  819     4842  1 !--
;  820     4843  1 ROUTINE REMASTER_CONTROL ( LOC_UCB: REF BLOCK [,BYTE], LKSB: VECTOR [2,LONG] ) =
;  821     4844  2 BEGIN
;  822     4845  2 !
;  823     4846  2 ! Remastering a control journal.
;  824     4847  2 !
;  825     4848  2 !
;  826     4849  2 ! Get ADBs from Allocated Device List for this control journal and, if the
;  827     4850  2 ! device indicated is "not allocated" and "not mounted", issue an $ALLJDR
;  828     4851  2 ! service to allocate the journal device.
;  829     4852  2 !
;  830     4853  2 ! As each ADB is processed, get rid of it by setting the flag ADB$V_PURGED.
;  831     4854  2 ! Here, "processed" means either successfully allocating and /or mounting, or
;  832     4855  2 ! discovering that somebody beat us to it. No locks are used at this level to
;  833     4856  2 ! arbitrate access to these devices, since that is handled properly by ALLOC
;  834     4857  2 ! and MOUNT anyway.
;  835     4858  2 !
;  836     4859  2 LOCAL
;  837     4860  2     ADL          : REF BLOCK [,BYTE],            ! allocated device list
;  838     4861  2     LOC_ADL      : REF BLOCK [,BYTE],            ! local copy of ADL
;  839     4862  2     ADB          : REF BLOCK [,BYTE],            ! current alloc dev block
;  840     4863  2     ADL_LINK,                                    ! addr of link from previous ADL
;  841     4864  2     UCB          : REF BLOCK [,BYTE],            ! addr of actual UCB
;  842     4865  2     DEVICE_NAME : BLOCK [8,BYTE]                 ! descriptor of device name
;  843     4866  2                   PRESET ( [DSC$W_LENGTH] = 0,
;  844     4867  2                            [DSC$B_DTYPE]  = 0,
;  845     4868  2                            [DSC$B_CLASS]  = 0),
;  846     4869  2     STATUS       : LONG;
;  847     4870  2
;  848     4871  2 LABEL
;  849     4872  2     ADB_LIST;                                    ! block defines linked ADBs
;  850     4873  2 !
;  851     4874  2 ! Lock the chain so we can get the ADL
;  852     4875  2 !
;  853     4876  2 RELOCK_CHAIN ( .LKSB );
;  854     4877  2 !
;  855     4878  2 ! Make a copy of this ADL and all extensions thereto, and
;  856     4879  2 ! mark all the ADBs in the list "purged" while we're about
;  857     4880  2 ! it. They will be reused if and when we manage to allocate
;  858     4881  2 ! some of these devices.
;  859     4882  2 !
```

```
;  860          4883  2 UCB = .LOC_UCB [UCB$L_LINK];
;  861          4884  2 ADL = .UCB [UCB$L_JNL_ADL];
;  862          4885  2 ADL_LINK = LOC_UCB [UCB$L_JNL_ADL];
;  863          4886  2 WHILE 1 DO
;  864          4887  3     BEGIN
;  865          4888  3     !
;  866          4889  3     ! Process single ADL...copy the whole thing into a
;  867          4890  3     ! buffer and point to it from our copy of the UCB.
;  868          4891  3     !
;  869          4892  3     STATUS = EXE$ALONONPAGED ( .ADL [ADL$W_SIZE], LOC_ADL );
;  870          4893  3     CH$MOVE ( .ADL [ADL$W_SIZE], .ADL, .LOC_ADL );
;  871          4894  3     !
;  872          4895  3     ! If first ADL, set up pointer from the UCB.
;  873          4896  3     ! If multiple ADLs, set up link in the duplicate copy.
;  874          4897  3     !
;  875          4898  3     .ADL_LINK = .LOC_ADL;
;  876          4899  3     !
;  877          4900  3     ! Mark all ADBs for this ADL available.
;  878          4901  3     !
;  879          4902  3     ADB = .ADL + .ADL [ADL$W_FSTADB];
;  880          4903  3     INCR COUNT FROM 1 TO .ADL [ADL$W_ADBCNT]
;  881          4904  3     DO
;  882          4905  4         BEGIN
;  883          4906  4         ADB [ADB$V_PURGED] = 1;
;  884          4907  4         ADB = .ADB + ADB$K_LENGTH;
;  885          4908  3         END;
;  886          4909  3     !
;  887          4910  3     ! Next ADL (if any)...
;  888          4911  3     !
;  889          4912  3     IF (ADL = .ADL [ADL$L_LINK]) EQL 0
;  890          4913  3     THEN
;  891          4914  3         !
;  892          4915  3         ! If no extension, we're done
;  893          4916  3         !
;  894          4917  3         EXITLOOP
;  895          4918  3     ELSE
;  896          4919  3         !
;  897          4920  3         ! Repeat the process with the next ADL. Keep track
;  898          4921  3         ! of where the forward link needs to go.
;  899          4922  3         !
;  900          4923  3         ADL_LINK = LOC_ADL [ADL$L_LINK];
;  901          4924  2     END;                                ! WHILE 1 DO
;  902          4925  2 !
;  903          4926  2 ! At this point we have made a copy of each ADL for this control journal, and
;  904          4927  2 ! linked them to our copy of that UCB. We have also marked all the ADBs for
;  905          4928  2 ! this control journal as unused (purged) so they can be reused. We can release
;  906          4929  2 ! the UCB chain. Locking from here forward will be based on normal allocate
;  907          4930  2 ! and mount protocols.
;  908          4931  2 !
;  909          4932  2 UCB [UCB$V_JNL_UNMAST] = 0;
;  910          4933  2 FREE_CHAIN ( .[KSB );
;  911          4934  2 !
;  912          4935  2 ! Walk through our copies of the ADLs and reallocate/remount those devices
;  913          4936  2 ! listed.
;  914          4937  2 !
;  915          4938  2 LOC_ADL = .LOC_UCB [UCB$L_JNL_ADL];
;  916          4939  2 WHILE 1 DO
```

```
;  917          4940  3     BEGIN
;  918          4941  3     !
;  919          4942  3     ! Process one ADL.
;  920          4943  3     !
;  921          4944  3     ADB = .LOC_ADL + .LOC_ADL [ADL$W_FSTADB];
;  922          4945  3     INCR COUNT FROM 1 TO .LOC_ADL [ADL$W_ADBCNT]
;  923          4946  3     DO
;  924          4947  4         BEGIN
;  925          4948  4         !
;  926          4949  4         ! Process one ADB.
;  927          4950  4         !
;  928          4951  4         LOCAL
;  929          4952  4             ADB_STATUS          : LONG,         ! "got whole vol set" or nay
;  930          4953  4             FIRST_ADB           : LONG;         ! first ADB of vol set
;  931          4954  4
;  932          4955  4         FIRST_ADB = .ADB;
;  933          4956  4         ADB_STATUS = 1;                         ! assume success
;  934          4957  5         IF (.ADB [ADB$V_MOUNTED]) AND (NOT .ADB [ADB$V_PURGED])
;  935          4958  4         THEN
;  936          4959  4
;  937          4960  4 ADB_LIST:
;  938          4961  4
;  939          4962  5             BEGIN
;  940          4963  5             UNTIL .ADB EQL 0
;  941          4964  5             DO
;  942          4965  6                 BEGIN
;  943          4966  6                 !
;  944          4967  6                 ! This ADB is active. Allocate the device. If linked with
;  945          4968  6                 ! other ADBs, allocate them as well.
;  946          4969  6                 !
;  947          4970  6                 DEVICE_NAME [DSC$W_LENGTH] = .ADB [ADB$B_NAMELEN];
;  948          4971  6                 DEVICE_NAME [DSC$A_POINTER] = ADB [ADB$B_DEVNAM];
;  949          4972  6                 STATUS = CJF$ALLJDR ( DEVICE_NAME );
;  950          4973  6                 IF NOT .STATUS THEN LEAVE ADB_LIST WITH ADB_STATUS = 0;
;  951          4974  6                 ADB [ADB$V_PURGED] = 1;
;  952          4975  6                 !
;  953          4976  6                 ! Get next ADB in this volume set, if any.
;  954          4977  6                 !
;  955          4978  6                 ADB = .ADB [ADB$L_LINK];
;  956          4979  5                 END;
;  957          4980  4             END;                            ! ADB_LIST block
;  958          4981  4     !
;  959          4982  4     ! Done walking one list of linked ADBs. See if we were successful
;  960          4983  4     ! with all devices therein.
;  961          4984  4     !
;  962          4985  4     ADB = .FIRST_ADB;
;  963          4986  4     IF NOT .ADB_STATUS
;  964          4987  4     THEN
;  965          4988  4         !
;  966          4989  4         ! We couldn't allocate all devices in this ADB list. We should
;  967          4990  4         ! deallocate those which were allocated, so someone else has a
;  968          4991  4         ! chance. Those which were allocated successfully will have
;  969          4992  4         ! the PURGED bit set; we deallocate these, and set the PURGED bit
;  970          4993  4         ! in the rest.
;  971          4994  4         !
;  972          4995  4         UNTIL .ADB EQL 0
;  973          4996  4         DO
```

```
;  974          4997  5                    BEGIN
;  975          4998  5                    IF .ADB [ADB$V_PURGED]
;  976          4999  5                    THEN
;  977          5000  6                        BEGIN
;  978          5001  6                        DEVICE_NAME [DSC$W_LENGTH] = .ADB [ADB$B_NAMELEN];
;  979          5002  6                        DEVICE_NAME [DSC$A_POINTER] = ADB [ADB$B_DEVNAM];
;  980          5003  6                        CJF$DEALJDR ( DEVICE_NAME );
;  981          5004  6                        END
;  982          5005  5                    ELSE
;  983          5006  5                        ADB [ADB$V_PURGED] = 1;
;  984          5007  5                    ADB = .ADB [ADB$L_LINK];
;  985          5008  4                    END;
;  986          5009  4            !
;  987          5010  4            ! Next ADB.
;  988          5011  4            !
;  989          5012  4            ADB = .FIRST_ADB + ADB$K_LENGTH;
;  990          5013  3            END;                                    ! Process ADB
;  991          5014  3        !
;  992          5015  3        ! Proceed with the next ADL, if any.
;  993          5016  3        !
;  994          5017  3        IF (LOC_ADL = .LOC_ADL [ADL$L_LINK]) EQL 0 THEN EXITLOOP;
;  995          5018  2        END;                                        ! Process ADL
;  996          5019  2    !
;  997          5020  2    ! Done handling this control journal. Deallocate the memory we allocated for
;  998          5021  2    ! our copies of the ADLs.
;  999          5022  2    !
; 1000          5023  2    LOC_ADL = .LOC_UCB [UCB$L_JNL_ADL];
; 1001          5024  2    UNTIL .LOC_ADL EQL 0
; 1002          5025  2    DO
; 1003          5026  3        BEGIN
; 1004          5027  3        ADL = .LOC_ADL [ADL$L_LINK];
; 1005          5028  3        EXE$DEANONPAGED ( .LOC_ADL );
; 1006          5029  3        LOC_ADL = .ADL;
; 1007          5030  2        END;
; 1008          5031  2    !
; 1009          5032  2    ! Delete the control journal itself.
; 1010          5033  2    !
; 1011          5034  2    RELOCK_CHAIN ( .LKSB );
; 1012          5035  2    UCB = .LOC_UCB [UCB$L_LINK];
; 1013          5036  2    DELETE_IODB ( .UCB );
; 1014          5037  2    IF .UCB [UCB$W_REFC] NEQ 0
; 1015          5038  2    THEN
; 1016          5039  2        UCB [UCB$V_DELETEUCB] = 1
; 1017          5040  2    ELSE
; 1018          5041  2        DELETE_UCB ( .UCB );
; 1019          5042  2    FREE_CHAIN ( .LKSB );
; 1020          5043  2    !
; 1021          5044  2    ! That's all
; 1022          5045  2    !
; 1023          5046  2    RETURN 1;
; 1024          5047  1    END;                        ! Process control journal


                                                OFFC 00000 REMASTER_CONTROL:
```

```
                                                   .WORD    Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11   ; 4843
                   5E             0C  C2 00002     SUBL2    #12, SP                                ;
                          04      AE  7C 00005     CLRQ     DEVICE_NAME                            ; 4868
                          08      AC  DD 00008     PUSHL    LKSB                                   ; 4876
          0000V    CF             01  FB 0000B     CALLS    #1, RELOCK_CHAIN                       ;
                   58     04      AC  D0 00010     MOVL     LOC_UCB, R8                            ; 4883
                   5A     30      A8  D0 00014     MOVL     48(R8), UCB                            ;
                   57   00A0      CA  D0 00018     MOVL     160(UCB), ADL                          ; 4884
                   5B   00A0      C8  9E 0001D     MOVAB    160(R8), ADL_LINK                      ; 4885
                   51     08      A7  3C 00022 1$: MOVZWL   8(ADL), R1                             ; 4892
             00000000G            00  16 00026     JSB      EXE$ALONONPAGED                        ;
                   6E             50  D0 0002C     MOVL     R0, STATUS                             ;
                   59             52  D0 0002F     MOVL     R2, R9                                 ;
  69               67     08      A7  28 00032     MOVC3    8(ADL), (ADL), (LOC_ADL)               ; 4893
                   6B             59  D0 00037     MOVL     LOC_ADL, (ADL_LINK)                    ; 4898
                   56     14      A7  3C 0003A     MOVZWL   20(ADL), ADB                           ; 4902
                   56             57  C0 0003E     ADDL2    ADL, ADB                               ;
                   51     12      A7  3C 00041     MOVZWL   18(ADL), R1                            ; 4903
                                  50  D4 00045     CLRL     COUNT                                  ;
                                  07  11 00047     BRB      3$                                     ;
             04    A6             04  88 00049 2$: BISB2    #4, 4(ADB)                             ; 4906
                   56             1C  C0 0004D     ADDL2    #28, ADB                               ; 4907
  F5               50             51  F3 00050 3$: AOBLEQ   R1, COUNT, 2$                          ; 4903
                   57             67  D0 00054     MOVL     (ADL), ADL                             ; 4912
                                  05  13 00057     BEQL     4$                                     ;
                   5B             59  D0 00059     MOVL     LOC_ADL, ADL_LINK                      ; 4923
                                  C4  11 0005C     BRB      1$                                     ; 4886
             69    AA             02  8A 0005E 4$: BICB2    #2, 105(UCB)                           ; 4932
                          08      AC  DD 00062     PUSHL    LKSB                                   ; 4933
          0000V    CF             01  FB 00065     CALLS    #1, FREE_CHAIN                         ;
                   59   00A0      C8  D0 0006A     MOVL     160(R8), LOC_ADL                       ; 4938
                   56     14      A9  3C 0006F 5$: MOVZWL   20(LOC_ADL), ADB                       ; 4944
                   56             59  C0 00073     ADDL2    LOC_ADL, ADB                           ;
                   55     12      A9  3C 00076     MOVZWL   18(LOC_ADL), R5                        ; 4945
                                  54  D4 0007A     CLRL     COUNT                                  ;
                                  6B  11 0007C     BRB      14$                                    ;
                   53             56  D0 0007E 6$: MOVL     ADB, FIRST_ADB                         ; 4955
                   52             01  D0 00081     MOVL     #1, ADB_STATUS                         ; 4956
  30         04    A6             01  E1 00084     BBC      #1, 4(ADB), 9$                         ; 4957
  2B         04    A6             02  E0 00089     BBS      #2, 4(ADB), 9$                         ;
                                  56  D5 0008E 7$: TSTL     ADB                                    ; 4963
                                  27  13 00090     BEQL     9$                                     ;
             04    AE     0C      A6  9B 00092     MOVZBW   12(ADB), DEVICE_NAME                   ; 4970
             08    AE     0D      A6  9E 00097     MOVAB    13(R6), DEVICE_NAME+4                  ; 4971
                          04      AE  9F 0009C     PUSHAB   DEVICE_NAME                            ; 4972
      00000000G    00             01  FB 0009F     CALLS    #1, CJF$ALLJDR                         ;
                   6E             50  D0 000A6     MOVL     R0, STATUS                             ;
                   04             6E  E8 000A9     BLBS     STATUS, 8$                             ; 4973
                                  52  D4 000AC     CLRL     ADB_STATUS                             ;
                                  09  11 000AE     BRB      9$                                     ;
             04    A6             04  88 000B0 8$: BISB2    #4, 4(ADB)                             ; 4974
                   56             66  D0 000B4     MOVL     (ADB), ADB                             ; 4978
                                  D5  11 000B7     BRB      7$                                     ; 4963
                   56             53  D0 000B9 9$: MOVL     FIRST_ADB, ADB                         ; 4985
                   26             52  E8 000BC     BLBS     ADB_STATUS, 13$                        ; 4986
                                  24  13 000BF 10$: BEQL    13$                                    ; 4995
  16         04    A6             02  E1 000C1     BBC      #2, 4(ADB), 11$                        ; 4998
```

```
                    04   AE        0C   A6  9B 000C6          MOVZBW  12(ADB), DEVICE_NAME                ;5001
                    08   AE        0D   A6  9E 000CB          MOVAB   13(R6), DEVICE_NAME+4              ;5002
                                   C    AE  9F 000D0          PUSHAB  DEVICE_NAME                        ;5003
              00000000G  00             01  FB 000D3          CALLS   #1, CJF$DEALJDR                    ;
                                        04  11 000DA          BRB     12$                                ;4998
                    04   A6             04  88 000DC 11$:     BISB2   #4, 4(ADB)                         ;5006
                         56             66  D0 000E0 12$:     MOVL    (ADB), ADB                         ;5007
                                        DA  11 000E3          BRB     10$                                ;4995
                         56        1C   A3  9E 000E5 13$:     MOVAB   28(R3), ADB                        ;5012
                91       54             55  F3 000E9 14$:     AOBLEQ  R5, COUNT, 6$                      ;4945
                         59             69  D0 000ED          MOVL    (LOC_ADL), LOC_ADL                 ;5017
                                        03  13 000F0          BEQL    15$                                ;
                                    FF7A  31 000F2          BRW     5$                                 ;
                         59      00A0   C8  D0 000F5 15$:     MOVL    160(R8), LOC_ADL                   ;5023
                                        11  13 000FA 16$:     BEQL    17$                                ;5024
                         57             69  D0 000FC          MOVL    (LOC_ADL), ADL                     ;5027
                         50             59  D0 000FF          MOVL    LOC_ADL, R0                        ;5028
                              00000000G 00  16 00102          JSB     EXE$DEANONPAGED                    ;
                         59             57  D0 00108          MOVL    ADL, LOC_ADL                       ;5029
                                        ED  11 0010B          BRB     16$                                ;5024
                                   08   AC  DD 0010D 17$:     PUSHL   LKSB                               ;5034
                    0000V  CF             01  FB 00110          CALLS   #1, RELOCK_CHAIN                   ;
                         5A        30   A8  D0 00115          MOVL    48(R8), UCB                        ;5035
                                        5A  DD 00119          PUSHL   UCB                                ;5036
              00000000G  00             01  FB 0011B          CALLS   #1, DELETE_IODB                    ;
                                   5C   AA  B5 00122          TSTW    92(UCB)                            ;5037
                                        06  13 00125          BEQL    18$                                ;
                    66   AA             01  88 00127          BISB2   #1, 102(UCB)                       ;5039
                                        09  11 0012B          BRB     19$                                ;
                                        5A  DD 0012D 18$:     PUSHL   UCB                                ;5041
                00000000G  00             01  FB 0012F          CALLS   #1, DELETE_UCB                     ;
                                   08   AC  DD 00136 19$:     PUSHL   LKSB                               ;5042
                    0000V  CF             01  FB 00139          CALLS   #1, FREE_CHAIN                     ;
                         50             01  D0 0013E          MOVL    #1, R0                             ;5046
                                            04 00141          RET                                        ;5047

; Routine Size:  322 bytes,    Routine Base:  CJF$CODE + 051D
```

```
;1026     5048  1 %SBTTL  'TEST_DEVICE - test accessibility of journal file device'
;1027     5049  1 !++
;1028     5050  1 ! TEST_DEVICE
;1029     5051  1 !
;1030     5052  1 ! FUNCTIONAL DESCRIPTION:
;1031     5053  1 !       Determine if the device on which this journal's journal file
;1032     5054  1 !       exists can be accessed from this node. We do this be testing to
;1033     5055  1 !       see if we can do a GETDVI on the device.
;1034     5056  1 !       THIS ROUTINE RUNS IN KERNEL MODE.
;1035     5057  1 ! CALLING SEQUENCE:
;1036     5058  1 !       STATUS = TEST_DEVICE ( LOCAL_UCB );
;1037     5059  1 ! FORMAL PARAMETERS:
;1038     5060  1 !       LOCAL_UCB       = copy of the journal's actual UCB.
;1039     5061  1 ! IMPLICIT PARAMETERS:
;1040     5062  1 !       I/O data base
;1041     5063  1 !       JNLRM array pointed by the UCB.
;1042     5064  1 ! COMPLETION CODES:
;1043     5065  1 !       SS$_NORMAL if the device is accessible
;1044     5066  1 !--
;1045     5067  1 ROUTINE TEST_DEVICE ( LOC_UCB : REF BLOCK [,BYTE] ) =
;1046     5068  2 BEGIN
;1047     5069  2 FIELD ITEM_LIST =
;1048     5070  2     SET
;1049     5071  2     ITMLST$W_BUFLEN     = [0,0,16,0],           ! item list buffer length
;1050     5072  2     ITMLST$W_CODE       = [2,0,16,0],           ! item list item code
;1051     5073  2     ITMLST$A_POINTER    = [4,0,32,0],           ! item list buffer address
;1052     5074  2     ITMLST$A_RETLEN     = [8,0,32,0],           ! item list return length address
;1053     5075  2     ITMLST$L_TERMZERO   = [12,0,32,0]           ! terminating zero for item list
;1054     5076  2     TES;
;1055     5077  2 LOCAL
;1056     5078  2     DEVICE_NAME         : BLOCK [8,BYTE]        ! descriptor of device name
;1057     5079  2                           PRESET ( [DSC$W_LENGTH] = 0,
;1058     5080  2                                    [DSC$B_DTYPE]  = 0,
;1059     5081  2                                    [DSC$B_CLASS]  = 0),
;1060     5082  2     DEVICE_STRING       : BLOCK [16,BYTE],      ! actual device name string
;1061     5083  2     CLUSTER_DEVNAM_STR  : BLOCK [16,BYTE],      ! array to receive name from GETDVI
;1062     5084  2     CLUSTER_DEVNAM      : BLOCK [8,BYTE]        ! descriptor of returned name
;1063     5085  2                           PRESET ( [DSC$W_LENGTH] = 0,
;1064     5086  2                                    [DSC$B_DTYPE]  = 0,
;1065     5087  2                                    [DSC$B_CLASS]  = 0),
;1066     5088  2     ITMLST              : BLOCK [4*4,BYTE]      ! item list to receive cluster devnam
;1067     5089  2                           FIELD ( ITEM_LIST )
;1068     5090  2                           PRESET ( [ITMLST$W_BUFLEN]   = 16,
;1069     5091  2                                    [ITMLST$W_CODE]     = DVI$_FULLDEVNAM,
;1070     5092  2                                    [ITMLST$L_TERMZERO] = 0 ),
;1071     5093  2     JNLRM               : REF BLOCK [,BYTE],    ! addr of remaster block
;1072     5094  2     UCB                 : REF BLOCK [,BYTE],    ! actual UCB for this journal
;1073     5095  2     LOCAL_IOSB          : VECTOR [2,LONG],      ! local i/o status block
;1074     5096  2     STATUS;
;1075     5097  2
;1076     5098  2 CLUSTER_DEVNAM [DSC$A_POINTER] = CLUSTER_DEVNAM_STR;
;1077     5099  2 ITMLST [ITMLST$A_POINTER] = CLUSTER_DEVNAM_STR;
;1078     5100  2 ITMLST [ITMLST$A_RETLEN] = CLUSTER_DEVNAM;
;1079     5101  2 !
;1080     5102  2 ! First, we pick up a copy of the remaster block.
;1081     5103  2 !
;1082     5104  2 UCB = .LOC_UCB [UCB$L_LINK];
```

```
;  1083          5105  2 JNLRM = .UCB [UCB$L_JNL_RMBLK];
;  1084          5106  2 !
;  1085          5107  2 ! Now create a descriptor to the device name.
;  1086          5108  2 !
;  1087          5109  2 SELECTONE 1 OF
;  1088          5110  2     SET
;  1089          5111  2     [.JNLRM [JNLRM$V_DSKJNL] ]:
;  1090          5112  3         BEGIN
;  1091          5113  3         !
;  1092          5114  3         ! We have a disk journal.
;  1093          5115  3         !
;  1094          5116  3         LOCAL
;  1095          5117  3             JNLRM_DEVICE        : REF BLOCK [,BYTE];
;  1096          5118  3         JNLRM_DEVICE = .JNLRM + JNLRM$K_DSKJNLLST;
;  1097          5119  3         DEVICE_NAME [DSC$W_LENGTH] = .JNLRM_DEVICE [JNLRM$W_DEVNAMLEN];
;  1098          5120  3         DEVICE_NAME [DSC$A_POINTER] = .JNLRM + .JNLRM_DEVICE [JNLRM$W_DEVNAMOFF];
;  1099          5121  2         END;
;  1100          5122  2     [.JNLRM [JNLRM$V_TAPJNL] ]:
;  1101          5123  3         BEGIN
;  1102          5124  3         !
;  1103          5125  3         ! We have a tape journal.
;  1104          5126  3         !
;  1105          5127  3         MUMBLE();
;  1106          5128  3         END;
;  1107          5129  2     TES;
;  1108          5130  2 !
;  1109          5131  2 ! Get the information about the device.
;  1110          5132  2 !
;  1111        P 5133  2 STATUS = $GETDVIW( DEVNAM = DEVICE_NAME,
;  1112        P 5134  2                    ITMLST = ITMLST,
;  1113        P 5135  2                    IOSB   = LOCAL_IOSB,
;  1114          5136  2                    EFN    = .LOCAL_EFN );
;  1115          5137  2 STATUS = .LOCAL_IOSB;
;  1116          5138  2 !
;  1117          5139  2 ! Do we need to do anything with the name?
;  1118          5140  2 !
;  1119          5141  2 RETURN .STATUS;
;  1120          5142  1 END;


                                                .PSECT  CJF$PLIT,NOWRT,NOEXE,  SHR,  PIC,2

                                00E8  0010  00024 P.AAE:  .WORD   16, 232                                  ;
                                        00# 00028         .BYTE   0[8]                                     ;
                                  00000000  00030         .LONG   0                                        ;

                                                .EXTRN  SYS$GETDVIW

                                                .PSECT  CJF$CODE,NOWRT,  SHR,  PIC,2

                                      003C 00000 TEST_DEVICE:
                                                .WORD   Save R2,R3,R4,R5                                 ; 5067
                           5E       B8   AE  9E 00002         MOVAB   -72(SP), SP                           ;
                                    40   AE  7C 00006         CLRQ    DEVICE_NAME                           ; 5081
                                    18   AE  7C 00009         CLRQ    CLUSTER_DEVNAM                        ; 5087
                  08   AE 00000000'  EF        10  28 0000C         MOVC3   #16, P.AAE, ITMLST            ; 5092
```

```
                          1C   AE        20   AE  9E 00015          MOVAB   CLUSTER_DEVNAM_STR, CLUSTER_DEVNAM+4      ; 5098
                          0C   AE        20   AE  9E 0001A          MOVAB   CLUSTER_DEVNAM_STR, ITMLST+4              ; 5099
                          10   AE        18   AE  9E 0001F          MOVAB   CLUSTER_DEVNAM, ITMLST+8                  ; 5100
                               50        04   AC  D0 00024          MOVL    LOC_UCB, R0                               ; 5104
                               50        30   A0  D0 00028          MOVL    48(R0), UCB                               ;
                               51      0104   C0  D0 0002C          MOVL    260(UCB), JNLRM                           ; 5105
                               13        0C   A1  E9 00031          BLBC    12(JNLRM), 1$                             ; 5111
                               50        18   A1  9E 00035          MOVAB   24(R1), JNLRM_DEVICE                      ; 5118
                          40   AE        02   A0  B0 00039          MOVW    2(JNLRM_DEVICE), DEVICE_NAME              ; 5119
                               52             60  3C 0003E          MOVZWL  (JNLRM_DEVICE), R2                        ; 5120
                44   AE         51             52  C1 00041          ADDL3   R2, JNLRM, DEVICE_NAME+4                  ;
                               0A             11  00046          BRB     2$                                        ; 5109
                     05    0C   A1             01  E1 00048 1$:      BBC     #1, 12(JNLRM), 2$                         ; 5122
                      0000V  CF                 00  FB 0004D          CALLS   #0, MUMBLE                                ; 5127
                                                7E  7C 00052 2$:      CLRQ    -(SP)                                     ; 5136
                                                7E  D4 00054          CLRL    -(SP)                                     ;
                                           0C   AE  9F 00056          PUSHAB  LOCAL_IOSB                                ;
                                           18   AE  9F 00059          PUSHAB  ITMLST                                    ;
                                           54   AE  9F 0005C          PUSHAB  DEVICE_NAME                               ;
                                                7E  D4 0005F          CLRL    -(SP)                                     ;
                               7E 00000000'     EF  9A 00061          MOVZBL  LOCAL_EFN, -(SP)                          ;
                  00000000G    00               08  FB 00068          CALLS   #8, SYS$GETDVIW                           ;
                               50               6E  D0 0006F          MOVL    LOCAL_IOSB, STATUS                        ; 5137
                                                04  00072          RET                                               ; 5142

; Routine Size:  115 bytes,    Routine Base:  CJF$CODE + 065F
```

```
;  1122          5143  1 %SBTTL  'RELOCK_CHAIN - convert chain lock to exclusive'
;  1123          5144  1 !++
;  1124          5145  1 ! RELOCK_CHAIN
;  1125          5146  1 !
;  1126          5147  1 ! FUNCTIONAL DESCRIPTION:
;  1127          5148  1 !       Common routine to convert the lock on the UCB chain indicated
;  1128          5149  1 !       by the accompanying lock status block from Null to Exclusive mode.
;  1129          5150  1 !               *** NOTE ***
;  1130          5151  1 !       THIS ROUTINE RUNS IN KERNEL MODE.
;  1131          5152  1 ! CALLING SEQUENCE:
;  1132          5153  1 !       RELOCK_CHAIN ( LKSB );
;  1133          5154  1 ! FORMAL PARAMETERS:
;  1134          5155  1 !       LKSB    = address of lock status block for this UCB chain.
;  1135          5156  1 ! COMPLETION CODES:
;  1136          5157  1 !
;  1137          5158  1 !--
;  1138          5159  1 ROUTINE RELOCK_CHAIN ( LKSB: VECTOR [2,LONG] ): NOVALUE =
;  1139          5160  2 BEGIN
;  1140        P 5161  2 $ENQW ( LKMODE = LCK$K_EXMODE,          ! exclusive lock
;  1141        P 5162  2         LKSB   = .LKSB,                 ! specify which lock
;  1142        P 5163  2         FLAGS  = LCK$M_CONVERT,         ! lock conversion
;  1143          5164  2         EFN    = .LOCAL_EFN );
;  1144          5165  2 RETURN;
;  1145          5166  1 END;


                                                        .EXTRN  SYS$ENQW

                                0000 00000 RELOCK_CHAIN:
                                           .WORD   Save nothing                         ;  5159
                             7E  7C 00002          CLRQ    -(SP)                       ;  5164
                             7E  7C 00004          CLRQ    -(SP)                       ;
                             7E  7C 00006          CLRQ    -(SP)                       ;
                          7E     02  7D 00008          MOVQ    #2, -(SP)               ;
                       04        AC  DD 0000B          PUSHL   LKSB                    ;
                                 05  DD 0000E          PUSHL   #5                      ;
                          7E 00000000'   EF  9A 00010          MOVZBL  LOCAL_EFN, -(SP)    ;
                 00000000G  00           0B  FB 00017          CALLS   #11, SYS$ENQW       ;
                                             04 0001E          RET                         ;  5166

; Routine Size:  31 bytes,    Routine Base:  $JF$CODE + 06D2
```

```
;  1147          5167  1 %SBTTL  'FREE_CHAIN'
;  1148          5168  1 !++
;  1149          5169  1 ! FREE_CHAIN
;  1150          5170  1 !
;  1151          5171  1 ! FUNCTIONAL DESCRIPTION:
;  1152          5172  1 !       Converts lock indicated by LKSB to null.
;  1153          5173  1 !               *** NOTE ***
;  1154          5174  1 !       THIS ROUTINE RUNS IN KERNEL MODE.
;  1155          5175  1 ! CALLING SEQUENCE:
;  1156          5176  1 !       FREE_CHAIN ( LKSB );
;  1157          5177  1 ! FORMAL PARAMETERS:
;  1158          5178  1 !       LKSB    = address of lock status block for the indicated lock.
;  1159          5179  1 ! COMPLETION CODES:
;  1160          5180  1 !
;  1161          5181  1 !--
;  1162          5182  1 ROUTINE FREE_CHAIN ( LKSB: VECTOR [2,LONG] ): NOVALUE =
;  1163          5183  2 BEGIN
;  1164        P 5184  2 IF NOT $ENQ ( LKMODE    = LCK$K_NLMODE,
;  1165        P 5185  2               LKSB      = .LKSB,
;  1166          5186  3               FLAGS     = LCK$M_NOQUEUE OR LCK$M_CONVERT )
;  1167          5187  2 THEN
;  1168          5188  2     MUMBLE ();
;  1169          5189  2 RETURN;
;  1170          5190  1 END;


                                0000 00000 FREE_CHAIN:
                                           .WORD   Save nothing                    ;  5182
                                7E  7C 00002       CLRQ    -(SP)                   ;  5186
                                7E  7C 00004       CLRQ    -(SP)                   ;
                                7E  7C 00006       CLRQ    -(SP)                   ;
                             7E    06  7D 00008       MOVQ    #6, -(SP)           ;
                          04       AC  DD 0000B       PUSHL   LKSB                ;
                                7E  7C 0000E       CLRQ    -(SP)                   ;
                00000000G     00    0B  FB 00010       CALLS   #11, SYS$ENQ        ;
                              05    50  E8 00017       BLBS    R0, 1$              ;
                    0000V     CF    00  FB 0001A       CALLS   #0, MUMBLE          ;  5188
                                    04 0001F 1$:       RET                         ;  5190

; Routine Size:  32 bytes,    Routine Base:  CJF$CODE + 06F1
```

```
;  1172          5191  1 %SBTTL  'RESTART_BASE - reschedule suspended base thread'
;  1173          5192  1 !++
;  1174          5193  1 ! RESTART_BASE
;  1175          5194  1 !
;  1176          5195  1 ! FUNCTIONAL DESCRIPTION:
;  1177          5196  1 !       If this is the Nth pass through this routine for the
;  1178          5197  1 !       indicated mutex, reschedule the indicated base context.
;  1179          5198  1 ! CALLING SEQUENCE:
;  1180          5199  1 !       RESTART_BASE ( COUNTER, CONTEXT, N )
;  1181          5200  1 ! FORMAL PARAMETERS:
;  1182          5201  1 !       COUNTER = address of allocated byte to count passes
;  1183          5202  1 !       CONTEXT = address of context block to reschedule if this is Nth pass
;  1184          5203  1 !       N       = number of passes before rescheduling CONTEXT
;  1185          5204  1 ! COMPLETION CODES:
;  1186          5205  1 !
;  1187          5206  1 !--
;  1188          5207  1 ROUTINE RESTART_BASE ( COUNTER, CONTEXT, N ): NOVALUE =
;  1189          5208  2 BEGIN
;  1190          5209  2 LOCAL
;  1191          5210  2     OWN_CONTEXT;
;  1192          5211  2 BUILTIN
;  1193          5212  2     INSQUE,REMQUE;
;  1194          5213  2 EXTERNAL
;  1195          5214  2     CSP$GQ_RESUME;
;  1196          5215  2
;  1197          5216  2 .COUNTER = ..COUNTER + 1;
;  1198          5217  2 IF ..COUNTER EQL N
;  1199          5218  2 THEN
;  1200          5219  3     BEGIN
;  1201          5220  3     REMQUE ( ..CONTEXT, OWN_CONTEXT );
;  1202          5221  3     INSQUE ( .OWN_CONTEXT, .(CSP$GQ_RESUME+4) );
;  1203          5222  2     END;
;  1204          5223  2 RETURN
;  1205          5224  1 END;


                                        0000 00000 RESTART_BASE:
                                                   .WORD   Save nothing                        ; 5207
                                    04  BC  D6 00002         INCL    @COUNTER                   ; 5216
                               50   0C  AC  9E 00005         MOVAB   N, R0                      ; 5217
                               50   04  BC  D1 00009         CMPL    @COUNTER, R0               ;
                                    13  12 0000D            BNEQ    1$                         ;
                               51   08  BC  9E 0000F         MOVAB   @CONTEXT, R1               ; 5220
                               50   00  B1  0F 00013         REMQUE  @0(R1), OWN_CONTEXT        ;
                               51 00000000G 00  9E 00017     MOVAB   CSP$GQ_RESUME+4, R1        ; 5221
                          00   B1          60  0E 0001E      INSQUE  (OWN_CONTEXT), @0(R1)      ;
                                           04 00022 1$:     RET                                ; 5224

; Routine Size:  35 bytes,    Routine Base:  CJF$CODE + 0711
```

```
;  1207          5225  1 %SBTTL  'CSP$$END_CONTEXT '
;  1208          5226  1 !++
;  1209          5227  1 ! CSP$$END_CONTEXT
;  1210          5228  1 !
;  1211          5229  1 ! FUNCTIONAL DESCRIPTION:
;  1212          5230  1 !       Terminate current thread, return to scheduler.
;  1213          5231  1 ! CALLING SEQUENCE:
;  1214          5232  1 !       CSP$$END_CONTEXT
;  1215          5233  1 ! FORMAL PARAMETERS:
;  1216          5234  1 !       None
;  1217          5235  1 ! IMPLICIT PARAMETERS:
;  1218          5236  1 !       CSP$GL_CURCTX is address of current context block
;  1219          5237  1 !       CSP$GL_BASE_FP is address of scheduler's call frame
;  1220          5238  1 ! COMPLETION CODES:
;  1221          5239  1 !
;  1222          5240  1 !--
;  1223          5241  1 GLOBAL ROUTINE CSP$$END_CONTEXT =
;  1224          5242  2 BEGIN
;  1225          5243  2 LOCAL
;  1226          5244  2     CONTEXT;                                  ! address of own context
;  1227          5245  2 BUILTIN
;  1228          5246  2     FP;
;  1229          5247  2
;  1230          5248  2 CONTEXT = .CSP$GL_CURCTX;                    ! get own context block
;  1231          5249  2 CSP$GL_CURCTX = 0;                          ! no current context anymore
;  1232          5250  2 LIB$FREE_VM ( UPLIT(CLX$K_LENGTH), CONTEXT );  ! free the data structure
;  1233          5251  2 FP = .CSP$GL_BASE_FP;                       ! fake the stack...
;  1234          5252  2 RETURN SS$_NORMAL;                         ! ...and return to the scheduler's caller
;  1235          5253  2
;  1236          5254  1 END;


                                                .PSECT  CJF$PLIT,NOWRT,NOEXE,  SHR,  PIC,2

                                00000140  00034 P.AAF:  .LONG   320                                    ;

                                                .PSECT  CJF$CODE,NOWRT,  SHR,  PIC,2

                                    0004 00000          .ENTRY  CSP$$END_CONTEXT, Save R2               ; 5241
                          52 00000000G  00   9E 00002   MOVAB   CSP$GL_CURCTX, R2                       ;
                                        62   DD 00009   PUSHL   CSP$GL_CURCTX                         ; 5248
                                        62   D4 0000B   CLRL    CSP$GL_CURCTX                         ; 5249
                                        5E   DD 0000D   PUSHL   SP                                    ; 5250
                             00000000'  EF   9F 0000F   PUSHAB  P.AAF                                 ;
                 00000000G   00         02   FB 00015   CALLS   #2, LIB$FREE_VM                       ;
                             5D 00000000G  00   D0 0001C   MOVL    CSP$GL_BASE_FP, FP                 ; 5251
                             50            01   D0 00023   MOVL    #1, R0                             ; 5252
                                           04   00026   RET                                           ; 5254

; Routine Size:  39 bytes,    Routine Base:  CJF$CODE + 0734

; 1237          5255  1
```

```
;  1239     5256  1 %SBTTL  'CALL_RCP'
;  1240     5257  1 !++
;  1241     5258  1 ! CALL_RCP
;  1242     5259  1 !
;  1243     5260  1 ! FUNCTIONAL DESCRIPTION:
;  1244     5261  1 !       Call the RCP on behalf of one newly remastered Recovery Unit journal.
;  1245     5262  1 !       The RCP will reacquire locks for any Recovery Units on this journal for
;  1246     5263  1 !       nodes no longer in the VAXcluster.
;  1247     5264  1 !       THIS ROUTINE RUNS IN KERNEL MODE.
;  1248     5265  1 ! CALLING SEQUENCE:
;  1249     5266  1 !       CALL_RCP ( .UCB );
;  1250     5267  1 ! FORMAL PARAMETERS:
;  1251     5268  1 !       UCB     = address of local copy of UCB for the Recovery Unit journal
;  1252     5269  1 ! COMPLETION CODES:
;  1253     5270  1 !
;  1254     5271  1 !--
;  1255     5272  1 ROUTINE CALL_RCP ( UCB: REF BLOCK [,BYTE] ): NOVALUE =
;  1256     5273  2 BEGIN
;  1257     5274  2 LITERAL
;  1258     5275  2     DEVNAM_LEN  = 16;                                   ! length of devnam str
;  1259     5276  2 LOCAL
;  1260     5277  2     RODB        : BLOCK [RODB$K_LENGTH+4,BYTE],         ! Object descriptor
;  1261     5278  2     RODBA       : BLOCK [RODBA$K_LENGTH+4,BYTE],        ! Object attributes
;  1262     5279  2     DEVNAM_STR  : BLOCK [DEVNAM_LEN,BYTE],              ! Device name string
;  1263     5280  2     DEVNAM      : BLOCK [8,BYTE]                        ! Device name desc
;  1264     5281  2                   PRESET ( [DSC$W_LENGTH]  = DEVNAM_LEN,
;  1265     5282  2                            [DSC$B_DTYPE]   = 0,
;  1266     5283  2                            [DSC$B_CLASS]   = 0,
;  1267     5284  2                            [DSC$A_POINTER] = DEVNAM_STR ),
;  1268     5285  2     JNLNAM      : BLOCK [8,BYTE]                        ! Journal name desc
;  1269     5286  2                   PRESET ( [DSC$W_LENGTH] = 0,
;  1270     5287  2                            [DSC$B_DTYPE]  = 0,
;  1271     5288  2                            [DSC$B_CLASS]  = 0),
;  1272     5289  2     LOCAL_IOSB  : VECTOR [2,LONG];                      ! local I/O status block
;  1273     5290  2     STATUS      : LONG;
;  1274     5291  2 !
;  1275     5292  2 ! Initialize fixed fields.
;  1276     5293  2 !
;  1277     5294  2 RODB [RODB$B_TYPE] = RODB$K_RUJNL;
;  1278     5295  2 RODB [RODB$B_COUNT] = 1;
;  1279     5296  2 RODB [RODB$W_SIZE] = 0;
;  1280     5297  2 RODB [RODB$A_POINTER] = RODBA;
;  1281     5298  2 RODB + RODB$K_LENGTH = 0;
;  1282     5299  2 RODBA [RODBA$B_TYPE] = RODBA$K_RUJDEVNAM;
;  1283     5300  2 RODBA + RODBA$K_LENGTH = 0;
;  1284     5301  2 !
;  1285     5302  2 ! Create a descriptor of the journal's home device name. For this, we use
;  1286     5303  2 ! a common routine from CJF.
;  1287     5304  2 !
;  1288     5305  2 JNLNAM [DSC$W_LENGTH] = .UCB [UCB$T_JNL_NAM];
;  1289     5306  2 JNLNAM [DSC$A_POINTER] = UCB [UCB$B_JNL_NAM];
;  1290     5307  2 STATUS = CREATE_RUDEV ( JNLNAM, DEVNAM );
;  1291     5308  2 IF NOT .STATUS THEN RETURN .STATUS;
;  1292     5309  2 !
;  1293     5310  2 ! Fill in the device name in the RODBA
;  1294     5311  2 !
;  1295     5312  2 RODBA [RODBA$W_SIZE] = .DEVNAM [DSC$W_LENGTH];
```

```
;1296        5313  2 RODBA [RODBA$A_POINTER] = .DEVNAM [DSC$A_POINTER];
;1297        5314  2 !
;1298        5315  2 ! Call the RCP to process all failed Recovery Units from this RU journal.
;1299        5316  2 !
;1300        5317  2 STATUS = CJF$RECOVERW( CJF$M_FAILOVER,          ! func
;1301        5318  2                        RODB,                    ! object
;1302        5319  2                        0,                       ! fltlst
;1303        5320  2                        0,                       ! acmode
;1304        5321  2                        .LOCAL_EFN,              ! efn
;1305        5322  2                        LOCAL_IOSB );            ! IOSB
;1306        5323  2
;1307        5324  2 IF .STATUS THEN STATUS = .LOCAL_IOSB;
;1308        5325  2 RETURN  STATUS;
;1309        5326  1 END;


                                0000 00000 CALL_RCP:
                                           .WORD   Save nothing                                    ; 5272
                     5E    C0   AE  9E 00002      MOVAB   -64(SP), SP
                10   AE         10  D0 00006      MOVL    #16, DEVNAM                             ; 5284
                14   AE      18 AE  9E 0000A      MOVAB   DEVNAM_STR, DEVNAM+4
                           08   AE  7C 0000F      CLRQ    JNLNAM                                  ; 5288
                34   AE    0104   8F  3C 00012    MOVZWL  #260, RODB                              ; 5294
                38   AE      28   AE  9E 00018    MOVAB   RODBA, RODB+4                           ; 5297
                             3C   AE  D4 0001D    CLRL    RODB+8                                  ; 5298
                28   AE         06  90 00020      MOVB    #6, RODBA                               ; 5299
                             30   AE  D4 00024    CLRL    RODBA+8                                 ; 5300
                     50      04   AC  D0 00027    MOVL    UCB, R0                                 ; 5305
                08   AE    00B8   C0  9B 0002B    MOVZBW  184(R0), JNLNAM
                0C   AE    00B9   C0  9E 00031    MOVAB   185(R0), JNLNAM+4                       ; 5306
                             10   AE  9F 00037    PUSHAB  DEVNAM                                  ; 5307
                             0C   AE  9F 0003A    PUSHAB  JNLNAM
             00000000G       00       FB 0003D    CALLS   #2, CREATE_RUDEV
                             2B       50 E9 00044 BLBC    STATUS, 1$                              ; 5308
                   2A   AE      10   AF  B0 00047 MOVW    DEVNAM, RODBA+2                         ; 5312
                   2C   AE      14   AE  D0 0004C MOVL    DEVNAM+4, RODBA+4                       ; 5313
                                     SE  DD 00051 PUSHL   SP                                      ; 5317
                             7E 00000000'  EF  9A 00053 MOVZBL  LOCAL_EFN, -(SP)                  ; 5321
                                     7E  7C 00057 CLRQ    -(SP)                                   ; 5317
                                   44    AE  9F 0005C PUSHAB  RODB
                                04000000 8F  DD 0005F PUSHL   #67108864
             00000000G       00       FB 00065    CALLS   #6, CJF$RECOVERW
                             03       50 E9 0006C BLBC    STATUS, 1$                              ; 5324
                             50       6E D0 0006F MOVL    LOCAL_IOSB, STATUS
                                      04 00072 1$: RET                                            ; 5326

; Routine Size:  115 bytes,    Routine Base:  CJF$CODE + 075B
```

```
; 1311          5327  1 ROUTINE MUMBLE = RETURN 1;



                                    0000 00000 MUMBLE: .WORD    Save nothing                    ; 5327
                              50          01  D0 00002         MOVL    #1, R0                  ;
                                          04 00005         RET                                 ;

; Routine Size:  6 bytes,    Routine Base:  CJF$CODE + 07CE

; 1312          5328  1 ROUTINE RESUME_FAILOVER = RETURN 1;



                                    0000 00000 RESUME_FAILOVER:
                                               .WORD    Save nothing                    ; 5328
                              50          01  D0 00002         MOVL    #1, R0                  ;
                                          04 00005         RET                                 ;

; Routine Size:  6 bytes,    Routine Base:  CJF$CODE + 07D4
```

```
; 1314          5329  1 END
; 1315          5330  0 ELUDOM



;                                 PSECT SUMMARY
;
;       Name                         Bytes                        Attributes
;
;   CJF$OWN                             25  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;   CJF$PLIT                            56  NOVEC,NOWRT,  RD ,NOEXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;   CJF$CODE                          2010  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)




;                           Library Statistics
;
;                                              -------- Symbols --------       Pages       Processing
;       File                                   Total    Loaded   Percent       Mapped      Time
;
;   _$255$DUA28:[SYSLIB]LIB.L32;1              18619        89         0       1000          00:01.4




;                                  COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:CSPCJFMAS/OBJ=OBJ$:CSPCJFMAS MSRC$:CSPCJFMAS/UPDATE=(ENH$:CSPCJFMAS)

; Size:          2010 code + 81 data bytes
; Run Time:         00:40.2
; Elapsed Time:     02:26.0
; Lines/CPU Min:      7965
; Lexemes/CPU-Min: 43192
; Memory Used:  426 pages
; Compilation Complete
```

0393 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

CSPCALL.
LIS

CSPCJFMAS.
LIS

CSP
LIS

CSPBRKTHR.
LIS

CONUTIL.
LIS

CSPCALLACT.
LIS

CONSUBS.
LIS

CSPOPCOM.LIS

CSPWAIT.LIS

CSPRCPCAC.LIS

CSPCJFRES.LIS

CSPQUORUM.LIS

DISTRLKI.LIS

CSPMOUNT.LIS

CSPVECTOR.LIS

CSPCLIENT.LIS

DSTRLOCK.LIS

DSTRLOCK.LIS