H 5

```
FFFFFFFFFFFFFFF       111          111    XXX         XXX
FFFFFFFFFFFFFFF       111          111    XXX         XXX
FFFFFFFFFFFFFFF       111          111    XXX         XXX
FFF                111111       111111    XXX         XXX
FFF                111111       111111    XXX         XXX
FFF                111111       111111    XXX         XXX
FFF                   111          111       XXX   XXX
FFF                   111          111       XXX   XXX
FFF                   111          111       XXX   XXX
FFFFFFFF.FFF          111          111          XXX
FFFFFFFFFFFF          111          111          XXX
FFFFFFFFFFFF          111          111          XXX
FFF                   111          111       XXX   XXX
FFF                   111          111       XXX   XXX
FFF                   111          111       XXX   XXX
FFF                   111          111    XXX         XXX
FFF                   111          111    XXX         XXX
FFF                   111          111    XXX         XXX
FFF                111111111    111111111 XXX         XXX
FFF                111111111    111111111 XXX         XXX
FFF                111111111    111111111 XXX         XXX
```

```
_$25
Symb
----
IOC$
IO_C
IO_C
IO_D
IO_F
IO_S
KILL

KILL
KILL
LB_E
LB_D
LB_F
LB_H
LB_L
LOCA
LOCA
LOCK

LOCK
LOCK
LOCK
LOC_
LOC_
L_CC
L_CC
L_DA
L_DA
MAIN
MAKE
MAKE
MAKE
MAKE
MAKE
MAKE

MAKE
MAKE
MAP_
MAP_

MAP_
MARK
MARK
MARK
MARK
MARK
```

```
RRRRRRRR    WW      WW  VV      VV  BBBBBBBB
RRRRRRRR    WW      WW  VV      VV  BBBBBBBB
RR      RR  WW      WW  VV      VV  BB      BB
RR      RR  WW      WW  VV      VV  BB      BB
RR      RR  WW      WW  VV      VV  BB      BB
RR      RR  WW      WW  VV      VV  BB      BB
RRRRRRRR    WW      WW  VV      VV  BBBBBBBB
RRRRRRRR    WW      WW  VV      VV  BBBBBBBB
RR  RR      WW  WW  WW  VV      VV  BB      BB
RR  RR      WW  WW  WW  VV      VV  BB      BB
RR    RR    WWWW  WWWW    VV  VV    BB      BB      ....
RR    RR    WWWW  WWWW    VV  VV    BB      BB      ....
RR      RR  WW      WW      VV      BBBBBBBB        ....
RR      RR  WW      WW      VV      BBBBBBBB        ....

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

```
;     1          0001  0 MODULE RWVB (
;     2          0002  0                 LANGUAGE (BLISS32),
;     3          0003  0                 IDENT = 'V04-000'
;     4          0004  0                 ) =
;     5          0005  1 BEGIN
;     6          0006  1
;     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 !
;    33          0033  1 ! FACILITY:  F11ACP Structure Level 1
;    34          0034  1 !
;    35          0035  1 ! ABSTRACT:
;    36          0036  1 !
;    37          0037  1 !       This routine performs the window turn necessary to map a
;    38          0038  1 !       virtual I/O transfer which is not mapped by the current
;    39          0039  1 !       window. It also receives virtual I/O errors for bad block
;    40          0040  1 !       processing.
;    41          0041  1 !
;    42          0042  1 ! ENVIRONMENT:
;    43          0043  1 !
;    44          0044  1 !       STARLET operating system, including privileged system services
;    45          0045  1 !       and internal exec routines.
;    46          0046  1 !
;    47          0047  1 !--
;    48          0048  1 !
;    49          0049  1 !
;    50          0050  1 ! AUTHOR:  Andrew C. Goldstein,  CREATION DATE:  7-Jan-1977  00:48
;    51          0051  1 !
;    52          0052  1 ! MODIFIED BY:
;    53          0053  1 !
;    54          0054  1 !       V03-006 CDS0005         Christian D. Saether     2-Mar-1984
;    55          0055  1 !               Add support for WRITE_TURN trapping.
;    56          0056  1 !
;    57          0057  1 !       V03-005 CDS0004         Christian D. Saether    26-Feb-1984
```

```
;    58          0058  1 !                 Call UNLOCK_XQP before requeueing the packet.
;    59          0059  1 !
;    60          0060  1 !       V03-004 CDS0003         Christian D. Saether    30-Dec-1983
;    61          0061  1 !                 Use L_NORM linkage and BIND_COMMON macro.
;    62          0062  1 !
;    63          0063  1 !       V03-003 CDS0002         Christian D. Saether    14-Sep-1983
;    64          0064  1 !                 Modify SERIAL_FILE interface.
;    65          0065  1 !
;    66          0066  1 !       V03-002 CDS0001         Christian D. Saether     5-May-1983
;    67          0067  1 !                 Add SERIAL_FILE call to synchronize xqp operations.
;    68          0068  1 !
;    69          0069  1 !       V03-001 ACG0320         Andrew C. Goldstein,    22-Mar-1983  12:44
;    70          0070  1 !                 Change byte count handling to track changes in IOPOST
;    71          0071  1 !
;    72          0072  1 !       V02-002 ACG0192         Andrew C. Goldstein,    18-Feb-1981  20:44
;    73          0073  1 !                 Fix attempt at bad block handling on write locked disk
;    74          0074  1 !
;    75          0075  1 !       V02-001 ACG0167         Andrew C. Goldstein,    16-Apr-1980  19:27
;    76          0076  1 !                 Previous revision history moved to F11B.REV
;    77          0077  1 !**
;    78          0078  1
;    79          0079  1
;    80          0080  1 LIBRARY 'SYS$LIBRARY:LIB.L32';
;    81          0081  1 REQUIRE 'SRC$:FCPDEF.B32';
;    82          1072  1
;    83          1073  1
;    84          1074  1 FORWARD ROUTINE
;    85          1075  1         READ_WRITEVB    : L_NORM,       ! main read/write virtual handling
;    86          1076  1         MARKBAD_FCB;                    ! mark bad block in FCB
```

```
;    88          1077  1 GLOBAL ROUTINE READ_WRITEVB  : L_NORM =
;    89          1078  1
;    90          1079  1 !++
;    91          1080  1 !
;    92          1081  1 ! FUNCTIONAL DESCRIPTION:
;    93          1082  1 !
;    94          1083  1 !       This routine performs the window turn necessary to map a
;    95          1084  1 !       virtual I/O transfer which is not mapped by the current
;    96          1085  1 !       window. It also receives virtual I/O errors for bad block
;    97          1086  1 !       processing. These are presently simply returned to the user.
;    98          1087  1 !
;    99          1088  1 ! CALLING SEQUENCE:
;   100          1089  1 !       READ_WRITEVB ()
;   101          1090  1 !
;   102          1091  1 ! INPUT PARAMETERS:
;   103          1092  1 !       NONE
;   104          1093  1 !
;   105          1094  1 ! IMPLICIT INPUTS:
;   106          1095  1 !       IO_PACKET: I/O packet of request
;   107          1096  1 !
;   108          1097  1 ! OUTPUT PARAMETERS:
;   109          1098  1 !       NONE
;   110          1099  1 !
;   111          1100  1 ! IMPLICIT OUTPUTS:
;   112          1101  1 !       NONE
;   113          1102  1 !
;   114          1103  1 ! ROUTINE VALUE:
;   115          1104  1 !       1 if request requeued to driver
;   116          1105  1 !       0 if error
;   117          1106  1 !
;   118          1107  1 ! SIDE EFFECTS:
;   119          1108  1 !       window turned
;   120          1109  1 !       request requeued to driver if mapped
;   121          1110  1 !
;   122          1111  1 !--
;   123          1112  1
;   124          1113  2 BEGIN
;   125          1114  2
;   126          1115  2 LOCAL
;   127          1116  2         PACKET          : REF BBLOCK,   ! pointer to I/O packet
;   128          1117  2         WINDOW          : REF BBLOCK,   ! file window
;   129          1118  2         FCB             : REF BBLOCK,   ! file FCB
;   130          1119  2         BLOCK_COUNT,                    ! number of blocks in transfer
;   131          1120  2         UNMAPPED,                       ! number of blocks not mapped
;   132          1121  2         MODE,                           ! mode (read/write) of transfer
;   133          1122  2         VBN,                            ! starting VBN of transfer
;   134          1123  2         LBN,                            ! translated LBN
;   135          1124  2         LAST_LBN;                       ! highest LBN touched by operation
;   136          1125  2
;   137          1126  2 BIND_COMMON;
;   138          1127  2
;   139          1128  2 EXTERNAL ROUTINE
;   140          1129  2         UNLOCK_XQP      : L_NORM,       ! unlock xqp and release cache
;   141          1130  2         SERIAL_FILE     : L_NORM,       ! serialize file operations
;   142          1131  2         ALLOCATION_LOCK : L_NORM,       ! get volume allocation lock
;   143          1132  2         MAP_VBN         : L_NORM,       ! map and turn window
;   144          1133  2         REQDEUE_REQ     : L_NORM,       ! requeue request to driver
```

```
;  145          1134  2          SCAN_BADLOG     : L_NORM;       ! scan bad block log file
;  146          1135  2
;  147          1136  2
;  148          1137  2 ! Extract the request parameters from the I/O packet. Compute VBN and LBN
;  149          1138  2 ! of the next block to be transferred.
;  150          1139  2 !
;  151          1140  2
;  152          1141  2 PACKET = .IO_PACKET;
;  153          1142  2 WINDOW = .PACKET[IRP$L_WIND];
;  154          1143  2 BLOCK_COUNT = (.PACKET[IRP$W_BCNT]+511) / 512;
;  155          1144  2 VBN = .PACKET[IRP$L_SEGVBN];
;  156          1145  2
;  157          1146  2 IF .VBN EQL 0 THEN ERR_EXIT (SS$_BADPARAM);
;  158          1147  2
;  159          1148  2 FCB = .WINDOW[WCB$L_FCB];
;  160          1149  2
;  161          1150  2 ! Serialize further processing on this file.
;  162          1151  2 !
;  163          1152  2
;  164          1153  2 PRIM_LCKINDX = SERIAL_FILE (FCB [FCB$W_FID]);
;  165          1154  2
;  166          1155  2 ! Attempt to map the request. If the map fails, report
;  167          1156  2 ! failure. Else requeue the request to the driver.
;  168          1157  2 !
;  169          1158  2
;  170          1159  2 LBN = MAP_VBN (.VBN, .WINDOW, .BLOCK_COUNT, UNMAPPED);
;  171          1160  2 IF .LBN EQL -1 THEN ERR_EXIT (SS$_ENDOFFILE);
;  172          1161  2
;  173          1162  2 IF .PACKET[IRP$V_VIRTUAL]
;  174          1163  2 THEN
;  175          1164  3     BEGIN
;  176          1165  3     LAST_LBN = .LBN + (.BLOCK_COUNT - .UNMAPPED - 1);
;  177          1166  3     IF .LBN GEQU .CURRENT_UCB[UCB$L_MAXBLOCK]
;  178          1167  3     OR .LAST_LBN GEQU .CURRENT_UCB[UCB$L_MAXBLOCK]
;  179          1168  3     THEN ERR_EXIT (SS$_ILLBLKNUM);
;  180          1169  3
;  181          1170  3     IF .WINDOW [WCB$V_WRITE_TURN]
;  182          1171  3     THEN
;  183          1172  3         IF .FCB [FCB$B_FID_NMX] NEQ 0
;  184          1173  3             OR .FCB [FCB$W_FID_NUM] GTRU 2
;  185          1174  3         THEN
;  186          1175  3
;  187          1176  3 ! This is not the index file or bitmap file, so we assume it
;  188          1177  3 ! is a directory.  Simply bump the data sequence number.  The
;  189          1178  3 ! unlock_xqp will store the updated value block.
;  190          1179  3 !
;  191          1180  3
;  192          1181  3             LB_DATASEQ [.PRIM_LCKINDX] = .LB_DATASEQ [.PRIM_LCKINDX] + 1
;  193          1182  3         ELSE
;  194          1183  4             BEGIN
;  195          1184  4             IF .FCB [FCB$W_FID_NUM] EQL 1
;  196          1185  4             THEN
;  197          1186  4
;  198          1187  4 ! This is the index file.  Determine if the VBN being written is within
;  199          1188  4 ! the index file bitmap or in the header area, and increment the appropriate
;  200          1189  4 ! sequence number to invalidate cached buffers.
;  201          1190  4 !
```

```
;  202          1191  4
;  203          1192  5                    BEGIN
;  204          1193  5                    LOCAL
;  205          1194  5                        HDRBASE;
;  206          1195  5
;  207          1196  5                    HDRBASE = .CURRENT_VCB [VCB$W_CLUSTER]*4
;  208          1197  5                            + .CURRENT_VCB [VCB$B_IBMAPSIZE];
;  209          1198  5
;  210          1199  5 ! Loop for all blocks being written.
;  211          1200  5 !
;  212          1201  5
;  213          1202  6                    INCR I FROM 0 TO (.BLOCK_COUNT - 1)
;  214          1203  5                    DO
;  215          1204  6                        BEGIN
;  216          1205  6                        LOCAL
;  217          1206  6                            FID     : BBLOCK [6];
;  218          1207  6
;  219          1208  6                        UNLOCK_XQP ();
;  220          1209  6
;  221          1210  6                        FID = .VBN + .I;
;  222          1211  6
;  223          1212  6                        IF .FID LEQU .HDRBASE
;  224          1213  6                        THEN
;  225          1214  6
;  226          1215  6 ! This is not within the header area.  Assume it is the index file bitmap.
;  227          1216  6 !
;  228          1217  6
;  229          1218  7                            BEGIN
;  230          1219  7                            ALLOCATION_LOCK ();
;  231          1220  7                            (LB_DATASEQ [0])<16,16> = .(LB_DATASEQ [0])<16,16> + 1;
;  232          1221  7                            END
;  233          1222  6                    ELSE
;  234          1223  6
;  235          1224  6 ! This is within the header area.  Calculate the file number and bump
;  236          1225  6 ! it's sequence number.  Note that extension headers are locked under
;  237          1226  6 ! the primary header's file number, and hence this does not directly
;  238          1227  6 ! invalidate them.  However, we will assume that whatever is out there
;  239          1228  6 ! mucking with headers in the first place will know enough to rewrite
;  240          1229  6 ! the primary header when screwing with extension headers and get them
;  241          1230  6 ! in that fashion.
;  242          1231  6 !
;  243          1232  6
;  244          1233  7                            BEGIN
;  245          1234  7                            LOCAL
;  246          1235  7                                LCKINDX;
;  247          1236  7
;  248          1237  7                            FID = .FID - .HDRBASE;
;  249          1238  7                            FID [FID$B_NMX] = .FID<16,8>;
;  250          1239  7                            FID [FID$B_RVN] = .CURRENT_RVN;
;  251          1240  7                            LCKINDX = SERIAL_FILE (FID);
;  252          1241  7                            LB_HDRSEQ [.LCKINDX] = .LB_HDRSEQ [.LCKINDX] + 1;
;  253          1242  6                            END;
;  254          1243  5                    END;                    ! of loop through all blocks
;  255          1244  5                END                         ! of this is file number 1
;  256          1245  4            ELSE
;  257          1246  4
;  258          1247  4 ! This is for file number 2, which is the storage bitmap.  Invalidate
```

```
;  259          1248  4 ! entries cached for it.
;  260          1249  4 !
;  261          1250  4
;  262          1251  5                 BEGIN
;  263          1252  5                 ALLOCATION_LOCK ();
;  264          1253  5                 (LB_DATASEQ [0])<0,16> = .(LB_DATASEQ [0])<0,16> + 1;
;  265          1254  4                 END;
;  266          1255  3             END;                            ! of either file 1 or 2.
;  267          1256  3
;  268          1257  3     UNLOCK_XQP ();
;  269          1258  3     KERNEL_CALL (REQUEUE_REQ, .PACKET, .LBN, .UNMAPPED);
;  270          1259  3     RETURN 1;
;  271          1260  3     END
;  272          1261  3
;  273          1262  3 ! If the virtual bit is not set, this is an I/O error on a file sent here
;  274          1263  3 ! for bad block processing. If the error is a parity, format, or datacheck
;  275          1264  3 ! error, we set the bad block bit in the FCB of the file and enter the
;  276          1265  3 ! block in question into the volume's bad block log. Note that we do not
;  277          1266  3 ! do this on errors on the volume's reserved files, which are not subject
;  278          1267  3 ! to dynamic bad block processing.
;  279          1268  3 !
;  280          1269  3
;  281          1270  2 ELSE
;  282          1271  3     BEGIN
;  283          1272  3     USER_STATUS[0] = .PACKET[IRP$L_IOST1];      ! get status to return to user
;  284          1273  3     USER_STATUS[1] = .PACKET[IRP$L_IOST2];
;  285          1274  3
;  286          1275  3     IF
;  287          1276  3         NOT .BBLOCK [CURRENT_UCB[UCB$L_DEVCHAR], DEV$V_SWL]
;  288          1277  4     AND (
;  289          1278  4             .(PACKET[IRP$L_IOST1])<0,16> EQL SS$_PARITY
;  290          1279  4         OR .(PACKET[IRP$L_IOST1])<0,16> EQL SS$_DATACHECK
;  291          1280  4         OR .(PACKET[IRP$L_IOST1])<0,16> EQL SS$_FORMAT
;  292          1281  4         )
;  293          1282  4     AND (
;  294          1283  4         .FCB[FCB$W_FID_NUM] GTRU .CURRENT_VCB[VCB$B_RESFILES]
;  295          1284  5         OR (.CURRENT_VCB[VCB$V_EXTFID]
;  296          1285  5             AND .FCB[FCB$B_FID_NMX] NEQ 0)
;  297          1286  4         )
;  298          1287  3     THEN
;  299          1288  4         BEGIN
;  300          1289  4         KERNEL_CALL (MARKBAD_FCB, .FCB);
;  301          1290  4         MODE = ENTER_READERR;                       assume read
;  302          1291  4         IF .PACKET[IRP$V_FCODE] EQL IO$_WRITEPBLK
;  303          1292  4         THEN MODE = ENTER_WRITERR;
;  304          1293  4         SCAN_BADLOG (FCB[FCB$W_FID], .VBN, .LBN, .MODE, 0);
;  305          1294  3         END;
;  306          1295  3     RETURN 0;
;  307          1296  2     END;
;  308          1297  2
;  309          1298  1 END;                                    ! end of routine READ_WRITEVB


                                                .TITLE  RWVB
                                                .IDENT  \V04-000\

                                                .EXTRN  UNLOCK_XQP, SERIAL_FILE
```

```
                                                        .EXTRN  ALLOCATION_LOCK
                                                        .EXTRN  MAP_VBN, REQUEUE_REQ
                                                        .EXTRN  SCAN_BADLOG

                                                        .PSECT  $CODE$,NOWRT,2

                                                03FC 00000      .ENTRY  READ_WRITEVB, Save R2,R3,R4,R5,R6,R7,R8,R9  ; 1077
                                         5E             0C   C2 00002      SUBL2   #12, SP                             ;
                                         59          80 AA   9E 00005      MOVAB   -128(BASE), R9                      ; 1124
                                         57        00A8 CA   9E 00009      MOVAB   168(BASE), R7                       ;
                                         54          90 AA   D0 0000E      MOVL    -112(BASE), PACKET                  ; 1141
                                         56          18 A4   D0 00012      MOVL    24(PACKET), WINDOW                  ; 1142
                                         50          32 A4   3C 00016      MOVZWL  50(PACKET), R0                      ; 1143
                                         50        01FF C0   9E 0001A      MOVAB   511(R0), R0                         ;
                                52       50  00000200 8F   C7 0001F      DIVL3   #512, R0, BLOCK_COUNT               ;
                                         58          48 A4   D0 00027      MOVL    72(PACKET), VBN                     ; 1144
                                                        03   12 0002B      BNEQ    1$                                  ; 1146
                                                        14   BF 0002D      CHMU    #20                                 ;
                                                             04 0002F      RET                                         ;
                                         53          18 A6   D0 00030 1$:  MOVL    24(WINDOW), FCB                     ; 1148
                                                     24 A3   9F 00034      PUSHAB  36(FCB)                             ; 1153
                                   0000G CF                  01   FB 00037      CALLS   #1, SERIAL_FILE                     ;
                                     18  AA                  50   D0 0003C      MOVL    R0, 24(BASE)                        ;
                                                   4004      8F   BB 00040      PUSHR   #^M<R2,SP>                          ; 1159
                                                             56   DD 00044      PUSHL   WINDOW                              ;
                                                             58   DD 00046      PUSHL   VBN                                 ;
                                   0000G CF                  04   FB 00048      CALLS   #4, MAP_VBN                         ;
                                         55                  50   D0 0004D      MOVL    R0, LBN                             ;
                                FFFFFFFF 8F                  55   D1 00050      CMPL    LBN, #-1                            ; 1160
                                                             05   12 00057      BNEQ    2$                                  ;
                                                   0870      8F   BF 00059      CHMU    #2160                               ;
                                                             04 0005D      RET                                         ;
                                03       2A A4                04   E0 0005E 2$:  BBS     #4, 42(PACKET), 3$                  ; 1162
                                                          00A8 31   00063      BRW     13$                                 ;
                                50                  52       6E   C3 00066 3$:  SUBL3   UNMAPPED, BLOCK_COUNT, R0        ; 1165
                                                    51    FF A045   9E 0006A      MOVAB   -1(R0)[LBN], LAST_LBN               ;
                                                    50    94   AA   D0 0006F      MOVL    -108(BASE), R0                      ; 1166
                                                00B0 C0           55   D1 00073      CMPL    LBN, 176(R0)                        ;
                                                                  07   1E 00078      BGEQU   4$                                  ;
                                                00B0 C0           51   D1 0007A      CMPL    LAST_LBN, 176(F                     ; 1167
                                                                  05   1F 0007F      BLSSU   5$                                  ;
                                                          00DC    8F   BF 00081 4$:  CHMU    #220                                ; 1168
                                                                  04 00085      RET                                         ;
                                71       15 A6                04   E1 00086 5$:  BBC     #4, 21(WINDOW), 12$                 ; 1170
                                                             29  A3   95 0008B      TSTB    41(F^B)                             ; 1172
                                                                 06   12 0008E      BNEQ    6$                                  ;
                                                       02    24  A3   B1 00090      CMPW    36(FCB), #2                         ; 1173
                                                                 09   1B 00094      BLEQU   7$                                  ;
                                                       50    18  AA   D0 00096 6$:  MOVL    24(BASE), R0                        ; 1181
                                                             6740 D6   0009A      INCL    (R7)[R0]                            ;
                                                                  5D   11 0009D      BRB     12$                                 ;
                                                       01    24  A3   B1 0009F 7$:  CMPW    36(FCB), #1                         ; 1184
                                                                 50   12 000A3      BNEQ    11$                                 ;
                                                       50    98  AA   D0 000A5      MOVL    -104(BASE), R0                      ; 1196
                                                       51    3C  A0   3C 000A9      MOVZWL  60(R0), R1                          ;
                                                       50    38  A0   9A 000AD      MOVZBL  56(R0), R0                          ; 1197
                                                       56        6041 DE   000B1      MOVAL   (R0)[R1], HDRBASE                   ;
```

```
                                59          01  CE 000B5          MNEGL   #1, I                                   ; 1202
                                            35  11 000B8          BRB     10$                                     ;
                          0000G CF          00  FB 000BA 8$:      CALLS   #0, UNLOCK_XQP                          ; 1208
                    04    AE    58          59  C1 000BF          ADDL3   I, VBN, FID                             ; 1210
                                56    04    AE  D1 000C4          CMPL    FID, HDRBASE                            ; 1212
                                            0A  1A 000C8          BGTRU   9$                                      ;
                          0000G CF          00  FB 000CA          CALLS   #0, ALLOCATION_LOCK                     ; 1219
                                      02    A7  B6 000CF          INCW    2(R7)                                   ; 1220
                                            1B  11 000D2          BRB     10$                                     ; 1212
                            04    AE        56  C2 000D4 9$:      SUBL2   HDRBASE, FID                            ; 1237
                            09    AE  06    AE  90 000D8          MOVB    FID+2, FID+5                            ; 1238
                            08    AE  A0    AA  90 000DD          MOVB    -96(BASE), FID+4                        ; 1239
                                      04    AE  9F 000E2          PUSHAB  FID                                     ; 1240
                          0000G CF          01  FB 000E5          CALLS   #1, SERIAL_FILE                         ;
                                0094  CA40  D6  000EA             INCL    148(BASE)[CKINDX]                       ; 1241
                    C7          59          52  F2 000EF 10$:     AOBLSS  R2, I, 8$                               ; 1202
                                            07  11 000F3          BRB     12$                                     ; 1184
                          0000G CF          00  FB 000F5 11$:     CALLS   #0, ALLOCATION_LOCK                     ; 1252
                                            67  B6 000FA          INCW    (R7)                                    ; 1253
                          0000G CF          00  FB 000FC 12$:     CALLS   #0, UNLOCK_XQP                          ; 1257
                                            6E  DD 00101          PUSHL   UNMAPPED                                ; 1258
                                            30  BB 00103          PUSHR   #^M<R4,R5>                              ;
                          0000G CF          03  FB 00105          CALLS   #3, REQUEUE_REQ                         ;
                                59          01  D0 0010A          MOVL    #1, R0                                  ; 1271
                                            04  00 0010D          RET                                             ;
                                69    38    A4  7D 0010E 13$:     MOVQ    56(PACKET), (R9)                        ; 1272
                                50    94    AA  D0 00112          MOVL    -108(BASE), R0                          ; 1276
                    55    3B    A0          01  E0 00116          BBS     #1, 59(R0), 17$                         ;
                  01F4    8F    38    A4  B1 0011B              CMPW    56(PACKET), #500                         ; 1278
                                            10  13 00121          BEQL    14$                                     ;
                    005C  8F    38    A4  B1 00123              CMPW    56(PACKET), #92                          ; 1279
                                            08  13 00129          BEQL    14$                                     ;
                    00BC  8F    38    A4  B1 0012B              CMPW    56(PACKET), #188                         ; 1280
                                            3D  12 00131          BNEQ    17$                                     ;
                                50    98    AA  D0 00133 14$:     MOVL    -104(BASE), R0                          ; 1283
                                51    4F    A0  9A 00137          MOVZBL  79(R0), R1                              ;
                      24        A3          51  B1 0013B          CMPW    R1, 36(FCB)                             ;
                                            0A  1F 0013F          BLSSU   15$                                     ;
                          2A    0B    A0        05  E1 00141          BBC     #5, 11(R0), 17$                         ; 1284
                                      29    A3  95 00146          TSTB    41(FCB)                                 ; 1285
                                            25  13 00149          BEQL    17$                                     ;
                                            53  DD 0014B 15$:     PUSHL   FCB                                     ; 1289
                          0000V CF          01  FB 0014D          CALLS   #1, MARKBAD_FCB                         ;
                                50          01  D0 00152          MOVL    #1, MODE                                ; 1290
                0B        20    A4          06  00 ED 00155          CMPZV   #0, #6, 32(PACKET), #11                 ; 1291
                                            03  12 0015B          BNEQ    16$                                     ;
                                50          02  D0 0015D          MOVL    #2, MODE                                ; 1292
                                            7E  D4 00160 16$:     CLRL    -(SP)                                   ; 1293
                                            50  DD 00162          PUSHL   MODE                                    ;
                                            55  DD 00164          PUSHL   LBN                                     ;
                                            58  DD 00166          PUSHL   VBN                                     ;
                                      24    A3  9F 00168          PUSHAB  36(FCB)                                 ;
                          0000G CF          05  FB 0016B          CALLS   #5, SCAN_BADLOG                         ;
                                            50  D4 00170 17$:     CLRL    R0                                      ; 1295
                                                04 00172          RET                                             ; 1298

; Routine Size:  371 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  311          1299  1 GLOBAL ROUTINE MARKBAD_FCB (FCB) =
;  312          1300  1
;  313          1301  1 !++
;  314          1302  1 !
;  315          1303  1 ! FUNCTIONAL DESCRIPTION:
;  316          1304  1 !
;  317          1305  1 !       This routine set the bad block bit in the indicated FCB.
;  318          1306  1 !
;  319          1307  1 !
;  320          1308  1 ! CALLING SEQUENCE:
;  321          1309  1 !       MARKBAD_FCB (ARG1)
;  322          1310  1 !
;  323          1311  1 ! INPUT PARAMETERS:
;  324          1312  1 !       ARG1: address of FCB
;  325          1313  1 !
;  326          1314  1 ! IMPLICIT INPUTS:
;  327          1315  1 !       NONE
;  328          1316  1 !
;  329          1317  1 ! OUTPUT PARAMETERS:
;  330          1318  1 !       NONE
;  331          1319  1 !
;  332          1320  1 ! IMPLICIT OUTPUTS:
;  333          1321  1 !       NONE
;  334          1322  1 !
;  335          1323  1 ! ROUTINE VALUE:
;  336          1324  1 !       1
;  337          1325  1 !
;  338          1326  1 ! SIDE EFFECTS:
;  339          1327  1 !       bad bit set in FCB
;  340          1328  1 !
;  341          1329  1 !--
;  342          1330  1
;  343          1331  2 BEGIN
;  344          1332  2
;  345          1333  2 MAP
;  346          1334  2         FCB             : REF BBLOCK;   ! FCB argument
;  347          1335  2
;  348          1336  2
;  349          1337  2 FCB[FCB$V_BADBLK] = 1;
;  350          1338  2
;  351          1339  2 RETURN 1;
;  352          1340  2
;  353          1341  1 END;                                    ! end of routine MARKBAD_FCB


                                        0000 00000      .ENTRY  MARKBAD_FCB, Save nothing       ; 1299
                              50    04  AC  D0 00002      MOVL    FCB, R0                         ; 1337
                           22 A0        04  88 00006      BISB2   #4, 34(R0)                      ;
                              50        01  D0 0000A      MOVL    #1, R0                          ; 1339
                                            04 0000D      RET                                     ; 1341

; Routine Size:  14 bytes,    Routine Base:  $CODE$ + 0173
```

```
;  354          1342  1
;  355          1343  1 END
;  356          1344  0 ELUDOM
```

```
;                                PSECT SUMMARY
;
;      Name                      Bytes                         Attributes
;
;  $CODE$                          385  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
```

```
;                              Library Statistics
;
;                                      -------- Symbols --------      Pages       Processing
;      File                            Total    Loaded   Percent      Mapped      Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1       18619        43         0      1000          00:02.0
```

```
;                                   COMMAND QUALIFIERS

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

; Size:          385 code + 0 data bytes
; Run Time:        00:21.2
; Elapsed Time:    00:47.6
; Lines/CPU Min:    3805
; Lexemes/CPU-Min: 44304
; Memory Used:  269 pages
; Compilation Complete
```

RWVB
LIS

RDBLOK
LIS

REQUEU
LIS

RWATTR
LIS

REMOVE
LIS

RDHEDR
LIS

RETDIR
LIS