M 1

```
BBBBBBBBBBBB        AAAAAAAAA        SSSSSSSSSSSS  RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBBBBBBBBBBB        AAAAAAAAA        SSSSSSSSSSSS  RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBBBBBBBBBBB        AAAAAAAAA        SSSSSSSSSSSS  RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBB         BBB  AAA         AAA  SSS              RRR         RRR        TTT        LLL
BBB         BBB  AAA         AAA  SSS              RRR         RRR        TTT        LLL
BBB         BBB  AAA         AAA  SSS              RRR         RRR        TTT        LLL
BBB         BBB  AAA         AAA  SSS              RRR         RRR        TTT        LLL
BBB         BBB  AAA         AAA  SSS              RRR         RRR        TTT        LLL
BBB         BBB  AAA         AAA  SSS              RRR         RRR        TTT        LLL
BBBBBBBBBBBB     AAA         AAA     SSSSSSSSS     RRRRRRRRRRRR           TTT        LLL
BBBBBBBBBBBB     AAA         AAA     SSSSSSSSS     RRRRRRRRRRRR           TTT        LLL
BBBBBBBBBBBB     AAA         AAA     SSSSSSSSS     RRRRRRRRRRRR           TTT        LLL
BBB         BBB  AAAAAAAAAAAAAAA              SSS  RRR   RRR              TTT        LLL
BBB         BBB  AAAAAAAAAAAAAAA              SSS  RRR   RRR              TTT        LLL
BBB         BBB  AAAAAAAAAAAAAAA              SSS  RRR   RRR              TTT        LLL
BBB         BBB  AAA         AAA              SSS  RRR      RRR           TTT        LLL
BBB         BBB  AAA         AAA              SSS  RRR      RRR           TTT        LLL
BBB         BBB  AAA         AAA              SSS  RRR      RRR           TTT        LLL
BBBBBBBBBBBB     AAA         AAA  SSSSSSSSSSSS     RRR         RRR        TTT        LLLLLLLLLLLLLLL
BBBBBBBBBBBB     AAA         AAA  SSSSSSSSSSSS     RRR         RRR        TTT        LLLLLLLLLLLLLLL
BBBBBBBBBBBB     AAA         AAA  SSSSSSSSSSSS     RRR         RRR        TTT        LLLLLLLLLLLLLLL
```

FILEID**BASVIRTIO   J 11

```
BBBBBBBB      AAAAAA      SSSSSSSS  VV      VV    IIIIII    RRRRRRRR    TTTTTTTTTT    IIIIII      OOOOOO
BBBBBBBB      AAAAAA      SSSSSSSS  VV      VV    IIIIII    RRRRRRRR    TTTTTTTTTT    IIIIII      OOOOOO
BB      BB  AA      AA  SS          VV      VV      II      RR      RR      TT          II      OO      OO
BB      BB  AA      AA  SS          VV      VV      II      RR      RR      TT          II      OO      OO
BB      BB  AA      AA  SS          VV      VV      II      RR      RR      TT          II      OO      OO
BB      BB  AA      AA  SS          VV      VV      II      RR      RR      TT          II      OO      OO
BBBBBBBB    AA      AA    SSSSSS    VV      VV      II      RRRRRRRR        TT          II      OO      OO
BBBBBBBB    AA      AA    SSSSSS    VV      VV      II      RRRRRRRR        TT          II      OO      OO
BB      BB  AAAAAAAAAA          SS  VV      VV      II      RR  RR          TT          II      OO      OO
BB      BB  AAAAAAAAAA          SS  VV      VV      II      RR  RR          TT          II      OO      OO
BB      BB  AA      AA          SS    VV  VV        II      RR    RR        TT          II      OO      OO    ....
BB      BB  AA      AA          SS    VV  VV        II      RR    RR        TT          II      OO      OO    ....
BBBBBBBB    AA      AA  SSSSSSSS        VV        IIIIII    RR      RR      TT        IIIIII      OOOOOO      ....
BBBBBBBB    AA      AA  SSSSSSSS        VV        IIIIII    RR      RR      TT        IIIIII      OOOOOO      ....

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
```

```
0 MODULE BAS$$VIRT_IO (                                 ! Virtual array I/O
0                 IDENT = '1-027'          ! File: BASVIRTIO.B32 Edit: MDL1027
0                 ) =
1 BEGIN
1 !
```

```
1 !
1
1 !++
1 ! FACILITY:  VAX-11 BASIC Virtual Array Support
1 !
1 ! ABSTRACT:
1 !
1 !       This module contains the I/O support for VAX-11 BASIC
1 !       virtual arrays.  In the context of the RTL these are called
1 !       "BASIC File Arrays", since they are not properly a part of
1 !       the VAX architecture.  This module comprises the UDF and REC
1 !       levels of I/O for this very simple I/O interface.
1 !
1 ! ENVIRONMENT:  VAX-11 User Mode
1 !
1 ! AUTHOR: John Sauter, CREATION DATE: 04-APR-1979
1 !
1 ! MODIFIED BY:
1 !
1 ! 1-001 - Original.  This version does no buffering.  It is just for
1 !           checking out the indexing routines.  JBS 04-APR-1979
1 ! 1-002 - Change BAS$$STOP to BAS$$STOP_IO wherever possible.
1 !           JBS 09-APR-1979
1 ! 1-003 - Improve comments based on DGP's review.  JBS 09-APR-1979
1 ! 1-004 - Recover from Record Stream Active RMS error.  JBS 09-APR-1979
1 !           JBS 09-APR-1979
1 ! 1-005 - Today (actually late last night) the compiler began producing
1 !           code for virtual arrays, so start debugging.  JBS 24-MAY-1979
1 ! 1-006 - Take the ALQ parameter out of the $FAB_INIT, so that FAB$L_ALQ
1 !           appears in the cross reference.  JBS 24-MAY-1979
```

```
1 ! 1-007 - Don't allocate if the file is already allocated beyond the
1 !           current record.  JBS 25-MAY-1979
1 ! 1-008 - Worry about two descriptors pointing to the same file.
1 !           JBS 11-JUN-1979
1 ! 1-009 - Remove the redundent DSC$ defintiions.  JBS 19-JUN-1979
1 ! 1-010 - POP I/O on error.  JBS 25-JUL-1979
1 ! 1-011 - The buffer size for a virtual array file must be 512 bytes.
1 !           JBS 20-AUG-1979
1 ! 1-012 - Correct a typo in BAS$VA_PURGE.  JBS 30-AUG-1979
1 ! 1-013 - Check for RMS$_RNF in parallel with RMS$_EOF.  JBS 17-SEP-1979
1 ! 1-014 - Disable EXTEND until it is fixed.  JBS 17-SEP-1979
1 ! 1-015 - When unwinding, mark that we have no buffer in memory, since
1 !           we want to retry all I/O operations.  JBS 17-SEP-1979
1 ! 1-016 - Signal errors if the RELEASE fails.  This will have to be
1 !           disabled to run under release 1.  JBS 17-SEP-1979
1 ! 1-017 - The VAH was bad design, because we cannot purge the virtual
1 !           arrays whenever we lose control (consider a divide by zero
1 !           under an ON ERROR GO BACK).  Therefore, remove VAH and do not
1 !           use the HANDLE field.  Also, put the code back to using release
1 !           1 RMS.  JBS 18-SEP-1979
1 ! 1-018 - Don't allow stores into virtual arrays opened read only.
1 !           JBS 07-NOV-1979
1 ! 1-019 - Convert to automatic record locking and NXR processing.
1 !           JBS 09-NOV-1979
1 ! 1-020 - Remove BAS$VA_PURGE, which has been a no-op since September 18,
1 !           since the compiler no longer refers to it.  JBS 26-NOV-1979
1 ! 1-021 - Don't call BAS$$CB_POP if the I/O data base has already been
1 !           cleaned up.  JBS 11-JUN-1980
1 ! 1-022 - Add new entry points to fetch/store entire virtual arrays
1 !         instead of individual array elements.  PLL 2-Apr-1982
1 ! 1-023 - Add support for decimal.  PLL 12-Apr-1982
1 ! 1-024 - Bug fix to entire array entry points - check to see if current
1 !          buffer needs to be written out before getting/putting a new
1 !          record.  PLL 3-May-1982
1 ! 1-025 - Bug fix to entire array entry points again.  If the array size
1 !          and the offset add up to less than 512, use the array size for
1 !          the initial number of bytes to copy.  PLL 10-May-1982
1 ! 1-026 - Make sure Record Access will always be by key in BAS$$VA_CLOSE.
1 !         DG 27-Mar-1984
1 ! 1-027 - check for RMS$_CONTROLC return status.  MDL 30-Mar-1984
1 !--
1
1 !<BLF/PAGE>
```

```
1 !
1 ! SWITCHES:
1 !
1
1 SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
1
1 !
1 ! LINKAGES:
1 !
1
1 REQUIRE 'RTLIN:OTSLNK';                               ! Define linkages
1
1 !
1 ! TABLE OF CONTENTS:
1 !
1
1 FORWARD ROUTINE
1     BAS$$VA_FETCH : NOVALUE,                          ! Fetch routine
1     BAS$$VA_STORE : NOVALUE,                          ! Store routine
1     BAS$$VA_CLOSE : CALL_CCB NOVALUE,                 ! CLOSE effector
1     BAS$$WHOLE_VA_FETCH : NOVALUE,                    ! Fetch whole array
1     BAS$$WHOLE_VA_STORE : NOVALUE,                    ! Store whole array
1     HANDLER;                                          ! POP CCB on UNWIND
1
1 !
1 ! INCLUDE FILES:
1 !
1
1 REQUIRE 'RTLML:OTSLUB';                               ! Get LUB definitions
1
1 REQUIRE 'RTLIN:BASIOERR';                             ! I/O error codes
1
1 REQUIRE 'RTLIN:RTLPSECT';                             ! Macros for defining psects
1
1 LIBRARY 'RTLSTARLE';                                  ! System symbols
1
1 !
1 ! MACROS:
1 !
1 !       NONE
1 !
1 ! EQUATED SYMBOLS:
1 !
1 !+
1 ! The following literal determines the span of the interlock on shared
1 ! files.  That is, the number of bytes which are interlocked after a
1 ! reference to a location in a virtual array.  This is also the buffer
1 ! size required on the OPEN for the file.
1 !-
1
1 LITERAL
1     K_BLOCK_LENGTH = 512;                             ! Number of bytes in a virtual block
1
1 !
1 ! PSECTS:
1 !
1 DECLARE_PSECTS (BAS);                                 ! Declare psects for BAS$ facility
```

```
;  159       0871  1 !
;  160       0872  1 ! OWN STORAGE:
;  161       0873  1 !
;  162       0874  1 !      NONE
;  163       0875  1 !
;  164       0876  1 ! EXTERNAL REFERENCES:
;  165       0877  1 !
;  166       0878  1
;  167       0879  1 EXTERNAL ROUTINE
;  168       0880  1     BAS$$STOP : NOVALUE,                          ! signals fatal error
;  169       0881  1     BAS$$CB_PUSH : JSB_CB_PUSH NOVALUE,           ! Load register CCB
;  170       0882  1     BAS$$CB_POP : JSB_CB_POP NOVALUE,             ! Done with register CCB
;  171       0883  1     BAS$$CB_GET : JSB_CB_GET NOVALUE,             ! Fetch current value of CCB
;  172       0884  1     BAS$$STOP_IO : NOVALUE,                       ! Signal fatal I/O error
;  173       0885  1     BAS$$SIGNAL_CTRLC : NOVALUE,                  ! Signal CTRL/C
;  174       0886  1     LIB$STOP : NOVALUE,                           ! Signal fatal error
;  175       0887  1     LIB$MATCH_COND;                               ! Match condition values
;  176       0888  1
;  177       0889  1 !+
;  178       0890  1 ! The following are the error codes used in this module.
;  179       0891  1 !-
;  180       0892  1
;  181       0893  1 EXTERNAL LITERAL
;  182       0894  1     BAS$K_VIRARROPE : UNSIGNED (8),               ! Virtual array not opened
;  183       0895  1     BAS$K_VIRARRDIS : UNSIGNED (8),               ! Virtual array not on disk
;  184       0896  1     BAS$K_VIRBUFTOO : UNSIGNED (8),               ! Virtual buffer too large
;  185       0897  1     BAS$K_ILLOPE : UNSIGNED (8),                  ! Illegal operation
;  186       0898  1     BAS$K_ILLILLACC : UNSIGNED (8),               ! Illegal or illogical access
;  187       0899  1     BAS$K_PROLOSSOR : UNSIGNED (8),               ! Program lost, sorry
;  188       0900  1     OTS$_FATINTERR;                               ! Fatal internal OTS error
;  189       0901  1
```

```
1 GLOBAL ROUTINE BAS$$VA_FETCH (                        ! Fetch routine
1           DESCRIP,                                    ! The descriptor for this virtual array
1           INDEX,                                      ! Linearized index
1           VALUE                                       ! Where to store array item
1     ) : NOVALUE =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !         Fetch a value from a virtual array.  Multiple bytes may be
1 !         fetched with a single call.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !         DESCRIP.mz.r    The descriptor for the virtual array
1 !         INDEX.rl.v      The byte offset into the array
1 !         VALUE.wz.r      The place to store the value.  The number of
1 !                         bytes to store is in the LENGTH field of
1 !                         DESCRIP.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !         NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !         NONE
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !         NONE
1 !
1 ! SIDE EFFECTS:
1 !
1 !         Signals if an error is encountered.
1 !
1 !--
1
2     BEGIN
2
2     MAP
2         DESCRIP : REF BLOCK [8, BYTE];
2
2     GLOBAL REGISTER
2         CCB = K_CCB_REG : REF BLOCK [, BYTE];
2
2     BUILTIN
2         ASHP;
2
2     LOCAL
2         CHAN,                                         ! The channel this array is defined on
2         HANDLE,                                       ! Pointer to info for this array
2         GET_STATUS,                                   ! Last RMS GET status
2         PUT_STATUS,                                   ! Last RMS PUT status
2         READ_RECORD,                                  ! 1 = we must read the record
2         SAVE_CCB : VOLATILE;                          ! CCB to POP, or zero.
```

```
2
2 !+
2 ! Establish a handler to pop the CCB when unwinding.
2 !-
2
2     ENABLE
2         HANDLER (SAVE_CCB);
2
2 !+
2 ! Fetch the array's channel number from its descriptor
2 !-
2     CHAN = .DESCRIP [DSC$L_LOGUNIT];
2 !+
2 ! Get a pointer to the LUB/ISB/RAB for this channel.  If the channel has not
2 ! been opened yet, this call will allocate the LUB/ISB/RAB, but we will reject
2 ! it for lack of the LUB$V_OPENED bit.
2 !-
2     BAS$$CB_PUSH (.CHAN, LUB$K_LUN_MIN);
2     SAVE_CCB = .CCB;
2
2     IF ( NOT .CCB [LUB$V_OPENED]) THEN BAS$$STOP (BAS$K_VIRARROPE);
2
2 !+
2 ! If the channel was not opened with organization VIRTUAL, reject it.  This
2 ! also catches channel 0, which is always open but never has VIRTUAL
2 ! organization.
2 !-
2
2     IF (.CCB [LUB$B_ORGAN] NEQ LUB$K_ORG_VIRTU) THEN BAS$$STOP_IO (BAS$K_VIRARRDIS);
2
2 !+
2 ! If this channel has been used for block I/O, reject it.
2 !-
2
2     IF (.CCB [LUB$V_BLK_USE]) THEN BAS$$STOP_IO (BAS$K_ILLOPE);
2
2 !+
2 ! If the record size declared for the file is not 512 bytes, reject it.
2 !-
2
2     IF (.CCB [RAB$W_USZ] NEQ K_BLOCK_LENGTH) THEN BAS$$STOP_IO (BAS$K_VIRBUFTOO);
2
2 !+
2 ! Mark the LUB as being used for a virtual array.
2 !-
2     CCB [LUB$V_VA_USE] = 1;
2 !+
2 ! Record access will always be by key
2 !-
2     CCB [RAB$B_RAC] = RAB$C_KEY;
2 !+
2 ! Mark the RAB so that a $GET to a non-existent record will still lock it.
2 !-
2     CCB [RAB$V_NXR] = 1;
2 !+
2 ! Set the address of our CLOSE appendage in the LUB.  If somebody else's
2 ! is already there, we have a serious problem.
```

```
2 !-
2
2       IF (.CCB [LUB$A_CLOSE] EQLA 0) THEN CCB [LUB$A_CLOSE] = BAS$$VA_CLOSE;
2
2       IF (.CCB [LUB$A_CLOSE] NEQA BAS$$VA_CLOSE) THEN BAS$$STOP_IO (BAS$K_PROLOSSOR);
2
2 !+
2 ! If this is not the first reference to this file, we may have to
2 ! write out the current buffer.  We will write only if the current buffer
2 ! is not the buffer we wish to access.  LUB$L_LOG_RECNO is initialized
2 ! to zero for virtual files.
2 !-
2
3       IF (.CCB [LUB$L_LOG_RECNO] EQL ((.INDEX + .DESCRIP [DSC$L_BYTEOFF])/K_BLOCK_LENGTH) + 1)
2       THEN
2           READ_RECORD = 0
2       ELSE
3           BEGIN
3 !+
3 ! We actually do the PUT only if the buffer has been changed since we last
3 ! read it, as recorded by LUB$V_OUTBUF_DR.
3 !-
3
4           IF (.CCB [LUB$V_OUTBUF_DR])
3           THEN
4               BEGIN
4               PUT_STATUS = $PUT (RAB = .CCB);
4
4           IF .PUT_STATUS EQL RMS$_CONTROLC
4           THEN
4               BAS$$SIGNAL_CTRLC ();
4
4 !+
4 ! If the PUT fails, we must worry about the RSA error, which can happen if
4 ! we are running at AST level, and the AST interrupted some RMS I/O.  If
4 ! we get this error, wait for it to go away.  Any other RMS error is fatal.
4 !-
4
5               IF ( NOT .PUT_STATUS)
4               THEN
5                   BEGIN
5
5                   WHILE (.PUT_STATUS EQL RMS$_RSA) DO
6                       BEGIN
6                       $WAIT (RAB = .CCB);
6                       PUT_STATUS = $PUT (RAB = .CCB);
6
6                       IF .PUT_STATUS EQL RMS$_CONTROLC
6                       THEN
6                           BAS$$SIGNAL_CTRLC ();
6
5                       END;
5
5                   IF ( NOT .PUT_STATUS) THEN BAS$$STOP_IO (BAS$K_IOERR_REC);
5
4                   END;
4
```

```
;  362          1073 4 !+
;  363          1074 4 ! The buffer is no longer "dirty", mark it so.
;  364          1075 4 !-
;  365          1076 4              CCB [LUB$V_OUTBUF_DR] = 0;
;  366          1077 3              END;
;  367          1078 3
;  368          1079 3          READ_RECORD = 1;
;  369          1080 2          END;
;  370          1081 2
;  371          1082 2 !+
;  372          1083 2 ! If necessary, read in the record containing the element we want.
;  373          1084 2 !-
;  374          1085 2
;  375          1086 3      IF (.READ_RECORD)
;  376          1087 2      THEN
;  377          1088 3          BEGIN
;  378          1089 3          CCB [LUB$L_LOG_RECNO] = ((.INDEX + .DESCRIP [DSC$L_BYTEOFF])/K_BLOCK_LENGTH) + 1;
;  379          1090 3          GET_STATUS = $GET (RAB = .CCB);
;  380          1091 3
;  381          1092 3          IF .GET_STATUS EQL RMS$_CONTROLC
;  382          1093 3          THEN
;  383          1094 3              BAS$$SIGNAL_CTRLC ();
;  384          1095 3
;  385          1096 3 !+
;  386          1097 3 ! If we get EOF, just clear the buffer.  This is compatable with
;  387          1098 3 ! the PDP-11.
;  388          1099 3 !-
;  389          1100 3
;  390          1101 4          IF ((.GET_STATUS EQL RMS$_EOF) OR (.GET_STATUS EQL RMS$_OK_RNF))
;  391          1102 3          THEN
;  392          1103 4              BEGIN
;  393          1104 4              CH$FILL (0, .CCB [RAB$W_USZ], .CCB [RAB$L_UBF]);
;  394          1105 4              CCB [RAB$L_RBF] = .CCB [RAB$L_UBF];
;  395          1106 4              CCB [RAB$W_RSZ] = .CCB [RAB$W_USZ];
;  396          1107 4              END
;  397          1108 3          ELSE
;  398          1109 4              BEGIN
;  399          1110 4
;  400          1111 5              IF ( NOT .GET_STATUS)
;  401          1112 4              THEN
;  402          1113 5                  BEGIN
;  403          1114 5 !+
;  404          1115 5 ! Again, worry about the RSA RMS error.
;  405          1116 5 !-
;  406          1117 5
;  407          1118 5                  WHILE (.GET_STATUS EQL RMS$_RSA) DO
;  408          1119 6                      BEGIN
;  409          1120 6                      $WAIT (RAB = .CCB);
;  410          1121 6                      GET_STATUS = $GET (RAB = .CCB);
;  411          1122 6
;  412          1123 6                      IF .GET_STATUS EQL RMS$_CONTROLC
;  413          1124 6                      THEN
;  414          1125 6                          BAS$$SIGNAL_CTRLC ();
;  415          1126 6
;  416          1127 5                      END;
;  417          1128 5
;  418          1129 5                  IF ( NOT .GET_STATUS) THEN BAS$$STOP_IO (BAS$K_IOERR_REC);
;  419        1130 5
;  420        1131 4                    END;
;  421        1132 4
;  422        1133 3                END;
;  423        1134 3
;  424        1135 2            END;
;  425        1136 2
;  426        1137 2  !+
;  427        1138 2  ! At this point, the proper record is in the buffer, and we can copy
;  428        1139 2  ! data from it.
;  429        1140 2  !-
;  430        1141 2      IF .DESCRIP [DSC$B_DTYPE] NEQ DSC$K_DTYPE_P
;  431        1142 2      THEN
;  432        1143 2          CH$MOVE (.DESCRIP [DSC$W_LENGTH],
;  433        1144 2              .CCB [RAB$L_RBF] + ((.INDEX + .DESCRIP [DSC$L_BYTEOFF]) MOD K_BLOCK_LENGTH), .VALUE)
;  434        1145 2      ELSE
;  435        1146 3          BEGIN
;  436        1147 3          MAP
;  437        1148 3              VALUE : REF BLOCK [12,BYTE],
;  438        1149 3              DESCRIP : REF BLOCK [12,BYTE];
;  439        1150 3          LOCAL
;  440        1151 3              COUNT;
;  441        1152 3          COUNT = .DESCRIP [DSC$B_SCALE] - .VALUE [DSC$B_SCALE];
;  442        1153 3          ASHP (COUNT, DESCRIP [DSC$W_LENGTH],
;  443        1154 3              .CCB [RAB$L_RBF] + ((.INDEX + .DESCRIP [DSC$L_BYTEOFF]) MOD K_BLOCK_LENGTH),
;  444        1155 3              %REF(0), VALUE [DSC$W_LENGTH], .VALUE [DSC$A_POINTER]);
;  445        1156 2          END;
;  446        1157 2
;  447        1158 2  !+
;  448        1159 2  ! Done with this I/O channel.
;  449        1160 2  !-
;  450        1161 2      BAS$$CB_POP ();
;  451        1162 1      END;                                                    ! end of BAS$$VA_FETCH


                                                                        .TITLE  BAS$$VIRT_IO
                                                                        .IDENT  \1-027\

                                                                        .EXTRN  BAS$$STOP, BAS$$CB_PUSH
                                                                        .EXTRN  BAS$$CB_POP, BAS$$CB_GET
                                                                        .EXTRN  BAS$$STOP_IO, BAS$$SIGNAL_CTRLC
                                                                        .EXTRN  LIB$STOP, LIB$MATCH_COND
                                                                        .EXTRN  BAS$K_VIRARROPE
                                                                        .EXTRN  BAS$K_VIRARRDIS
                                                                        .EXTRN  BAS$K_VIRBUFTOO
                                                                        .EXTRN  BAS$K_ILLOPE, BAS$K_ILLILLACC
                                                                        .EXTRN  BAS$K_PROLOSSOR
                                                                        .EXTRN  OTS$_FATINTERR, SYS$PUT
                                                                        .EXTRN  SYS$WAIT, SYS$GET

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

                                                 0FFC 00000             .ENTRY  BAS$$VA_FETCH, Save R2,R3,R4,R5,R6,R7,R8,-   ; 0902
                                                                                R9,R10,R11                                   ;
                                   5A 00000000G  00  9E 00002             MOVAB   BAS$$SIGNAL_CTRLC, R10                       ;
                                   59 00000000G  00  9E 00009             MOVAB   BAS$$STOP_IO, R9                             ;
                                                 7E  D4 00010             CLRL    SAVE_CCB                                     ; 0941
```

```
                     6D      01B4  CF  DE 00012       MOVAL   22$, (FP)                        ;
                     57        04  AC  D0 00017       MOVL    DESCRIP, R7                      ; 0970
                     52        FC  A7  D0 0001B       MOVL    -4(R7), CHAN                     ;
                                   50  D4 0001F       CLRL    R0                               ; 0976
                         00000000G 00  16 00021       JSB     BAS$$CB_PUSH                     ;
                     6E            5B  D0 00027       MOVL    CCB, SAVE_CCB                    ; 0977
                     0B        FC  AB  E8 0002A       BLBS    -4(CCB), 1$                      ; 0979
                     7E       00G  8F  9A 0002E       MOVZBL  #BAS$K_VIRARROPE, -(SP)          ;
            00000000G 00           01  FB 00032       CALLS   #1, BAS$$STOP                    ;
                     05        C4  AB  91 00039 1$:   CMPB    -60(CCB), #5                     ; 0987
                                   07  13 0003D       BEQL    2$                               ;
                     7E       00G  8F  9A 0003F       MOVZBL  #BAS$K_VIRARRDIS, -(SP)          ;
                     69            01  FB 00043       CALLS   #1, BAS$$STOP_IO                 ;
      07         FF  AB            01  E1 00046 2$:   BBC     #1, -1(CCB), 3$                  ; 0993
                     7E       00G  8F  9A 0004B       MOVZBL  #BAS$K_ILLOPE, -(SP)             ;
                     69            01  FB 0004F       CALLS   #1, BAS$$STOP_IO                 ;
              0200   8F        20  AB  B1 00052 3$:   CMPW    32(CCB), #512                    ; 0999
                                   07  13 00058       BEQL    4$                               ;
                     7E       00G  8F  9A 0005A       MOVZBL  #BAS$K_VIRBUFTOO, -(SP)          ;
                     69            01  FB 0005E       CALLS   #1, BAS$$STOP_IO                 ;
                 FF  AB            01  88 00061 4$:   BISB2   #1, -1(CCB)                      ; 1004
                 1E  AB            01  90 00065       MOVB    #1, 30(CCB)                      ; 1008
                 06  AB        80  8F  88 00069       BISB2   #128, 6(CCB)                     ; 1012
                               A4  AB  D5 0006E       TSTL    -92(CCB)                         ; 1018
                                   06  12 00071       BNEQ    5$                               ;
                 A4  AB     0000V  CF  9E 00073       MOVAB   BAS$$VA_CLOSE, -92(CCB)          ;
                     50     0000V  CF  9E 00079 5$:   MOVAB   BAS$$VA_CLOSE, R0                ; 1020
                     50        A4  AB  D1 0007E       CMPL    -92(CCB), R0                     ;
                                   07  13 00082       BEQL    6$                               ;
                     7E       00G  8F  9A 00084       MOVZBL  #BAS$K_PROLOSSOR, -(SP)          ;
                     69            01  FB 00088       CALLS   #1, BAS$$STOP_IO                 ;
      58         08  AC        F8  A7  C1 0008B 6$:   ADDL3   -8(R7), INDEX, R8                ; 1029
      50             58  00000200  8F  C7 00091       DIVL3   #512, R8, R0                     ;
                     53        01  A0  9E 00099       MOVAB   1(R0), R3                        ;
                     53        E0  AB  D1 0009D       CMPL    -32(CCB), R3                     ;
                                   04  12 000A1       BNEQ    7$                               ;
                                   54  D4 000A3       CLRL    READ_RECORD                      ; 1031
                                   5C  11 000A5       BRB     13$                              ;
      54         FE  AB            03  E1 000A7 7$:   BBC     #3, -2(CCB), 12$                 ; 1039
                                   5B  DD 000AC       PUSHL   CCB                              ; 1042
            00000000G 00           01  FB 000AE       CALLS   #1, SYS$PUT                      ;
                     52            50  D0 000B5       MOVL    R0, PUT_STATUS                   ;
             00010651 8F           52  D1 000B8       CMPL    PUT_STATUS, #67153               ; 1044
                                   03  12 000BF       BNEQ    8$                               ;
                     6A            00  FB 000C1       CALLS   #0, BAS$$SIGNAL_CTRLC            ; 1046
                     35            52  E8 000C4 8$:   BLBS    PUT_STATUS, 11$                  ; 1054
             000182DA 8F           52  D1 000C7 9$:   CMPL    PUT_STATUS, #99034               ; 1058
                                   23  12 000CE       BNEQ    10$                              ;
                                   5B  DD 000D0       PUSHL   CCB                              ; 1060
            00000000G 00           01  FB 000D2       CALLS   #1, SYS$WAIT                     ;
                                   5B  DD 000D9       PUSHL   CCB                              ; 1061
            00000000G 00           01  FB 000DB       CALLS   #1, SYS$PUT                      ;
                     52            50  D0 000E2       MOVL    R0, PUT_STATUS                   ;
             00010651 8F           52  D1 000E5       CMPL    PUT_STATUS, #67153               ; 1063
                                   D9  12 000EC       BNEQ    9$                               ;
                     6A            00  FB 000EE       CALLS   #0, BAS$$SIGNAL_CTRLC            ; 1065
                                   D4  11 000F1       BRB     9$                               ; 1058
```

```
                       06        52  E8 000F3 10$:    BLBS    PUT_STATUS, 11$                                   ; 1069
                       7E        01  CE 000F6         MNEGL   #1, -(SP)                                         ;
                       69        01  FB 000F9         CALLS   #1, BAS$$STOP_IO                                  ;
                 FE    AB        08  8A 000FC 11$:    BICB2   #8, -2(CCB)                                       ; 1076
                       54        01  D0 00100 12$:    MOVL    #1, READ_RECORD                                   ; 1079
                       7A        54  E9 00103 13$:    BLBC    READ_RECORD, 19$                                  ; 1086
                 E0    AB        53  D0 00106         MOVL    R3, -32(CCB)                                      ; 1089
                                 5B  DD 0010A         PUSHL   CCB                                               ; 1090
          00000000G    00        01  FB 0010C         CALLS   #1, SYS$GET                                       ;
                       56        50  D0 00113         MOVL    R0, GET_STATUS                                    ;
           00010651    8F        56  D1 00116         CMPL    GET_STATUS, #67153                                ; 1092
                                 03  12 0011D         BNEQ    14$                                               ;
                       6A        00  FB 0011F         CALLS   #0, BAS$$SIGNAL_CTRLC                             ; 1094
           0001827A    8F        56  D1 00122 14$:    CMPL    GET_STATUS, #98938                                ; 1101
                                 09  13 00129         BEQL    15$                                               ;
           00018049    8F        56  D1 0012B         CMPL    GET_STATUS, #98377                                ;
                                 14  12 00132         BNEQ    16$                                               ;
   20  AB          00  6E        00  2C 00134 15$:    MOVC5   #0, (SP), #0, 32(CCB), @36(CCB)                   ; 1104
                              24  BB    0013A                                                                   ;
                 28    AB     24  AB D0 0013C         MOVL    36(CCB), 40(CCB)                                  ; 1105
                 22    AB     20  AB B0 00141         MOVW    32(CCB), 34(CCB)                                  ; 1106
                                 38  11 00146         BRB     19$                                               ; 1101
                       35        56  E8 00148 16$:    BLBS    GET_STATUS, 19$                                   ; 1111
           000182DA    8F        56  D1 0014B 17$:    CMPL    GET_STATUS, #99034                                ; 1118
                                 23  12 00152         BNEQ    18$                                               ;
                                 5B  DD 00154         PUSHL   CCB                                               ; 1120
          00000000G    00        01  FB 00156         CALLS   #1, SYS$WAIT                                      ;
                                 5B  DD 0015D         PUSHL   CCB                                               ; 1121
          00000000G    00        01  FB 0015F         CALLS   #1, SYS$GET                                       ;
                       56        50  D0 00166         MOVL    R0, GET_STATUS                                    ;
           00010651    8F        56  D1 00169         CMPL    GET_STATUS, #67153                                ; 1123
                                 D9  12 00170         BNEQ    17$                                               ;
                       6A        00  FB 00172         CALLS   #0, BAS$$SIGNAL_CTRLC                             ; 1125
                                 D4  11 00175         BRB     17$                                               ; 1118
                       06        56  E8 00177 18$:    BLBS    GET_STATUS, 19$                                   ; 1129
                       7E        01  CE 0017A         MNEGL   #1, -(SP)                                         ;
                       69        01  FB 0017D         CALLS   #1, BAS$$STOP_IO                                  ;
                       56     0C  AC D0 00180 19$:    MOVL    VALUE, R6                                         ; 1144
                       15     02  A7 91 00184         CMPB    2(R7), #21                                        ; 1141
                                 16  13 00188         BEQL    20$                                               ;
       7E          00  58        01  7A 0018A         EMUL    #1, R8, #0, -(SP)                                 ; 1144
       50          50  8E 00000200 8F  7B 0018F       EDIV    #512, (SP)+, R0, R0                               ;
                   66     28 BB40 67  28 00198        MOVC3   (R7), @40(CCB)[R0], (R6)                          ;
                                 23  11 0019E         BRB     21$                                               ; 1143
                       51     08  A7 98 001A0 20$:    CVTBL   8(R7), COUNT                                      ; 1152
                       50     08  A6 98 001A4         CVTBL   8(R6), R0                                         ;
                       51        50  C2 001A8         SUBL2   R0, COUNT                                         ;
       7E          00  58        01  7A 001AB         EMUL    #1, R8, #0, -(SP)                                 ; 1154
       50          50  8E 00000200 8F  7B 001B0       EDIV    #512, (SP)+, R0, R0                               ;
       00     28 BB40  67        51  F8 001B9         ASHP    COUNT, (R7), @40(CCB)[R0], #0, (R6), @4(R6)      ; 1155
                 04    B6        66     001C0                                                                   ;
                    00000000G    00  16 001C3 21$:    JSB     BAS$$CB_POP                                       ; 1161
                                     04 001C9         RET                                                       ; 1162
                                  0000 001CA 22$:     .WORD   Save nothing                                      ; 0941
                       50     08  AC D0 001CC         MOVL    8(AP), R0                                         ;
                       50     04  A0 D0 001D0         MOVL    4(R0), R0                                         ;
                              FC  A0 9F 001D4         PUSHAB  SAVE_CCB                                          ;
```

```
                                   01   DD 001D7            PUSHL    #1                               ;
                                   5E   DD 001D9            PUSHL    SP                               ;
                     7E        04  AC   7D 001DB            MOVQ     4(AP), -(SP)                     ;
              0000V  CF            03   FB 001DF            CALLS    #3, HANDLER                      ;
                                        04 001E4            RET                                       ;

; Routine Size:  485 bytes,     Routine Base:  _BAS$CODE + 0000


;  452          1163  1
```

```
1 GLOBAL ROUTINE BAS$$VA_STORE (                        ! Store routine
1         DESCRIP,                                      ! The descriptor for this virtual array
1         INDEX,                                        ! Linearized index
1         VALUE                                         ! Where to find item for array
1     ) : NOVALUE =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Store a value in a virtual array.  Multiple bytes may be stored
1 !       with a single call.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       DESCRIP.mz.r    The descriptor for the virtual array
1 !       INDEX.rl.v      The byte offset into the array
1 !       VALUE.rz.r      The place from which to fetch the value.  The
1 !                       number of bytes to store is in the LENGTH field
1 !                       of DESCRIP.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !
1 !       Signals if an error is encountered.
1 !
1 !--
1
2     BEGIN
2
2     MAP
2         DESCRIP : REF BLOCK [8, BYTE];
2
2     GLOBAL REGISTER
2         CCB = K_CCB_REG : REF BLOCK [, BYTE];
2
2     BUILTIN
2         ASHP;
2
2     LOCAL
2         CHAN,                                         ! The channel this array is defined on
2         HANDLE,                                       ! Pointer to info for this array
2         GET_STATUS,                                   ! Last RMS GET status
2         PUT_STATUS,                                   ! Last RMS PUT status
2         READ_RECORD,                                  ! 1 = we must read the record
2         SAVE_CCB : VOLATILE;                          ! CCB to POP, or 0
```

```
!+
! Establish a handler to pop the CCB on unwind.
!-

    ENABLE
        HANDLER (SAVE_CCB);

!+
! Fetch the array's channel number from its descriptor
!-
    CHAN = .DESCRIP [DSC$L_LOGUNIT];
!+
! Get a pointer to the LUB/ISB/RAB for this channel.  If the channel has not
! been opened yet, this call will allocate the LUB/ISB/RAB, but we will reject
! it for lack of the LUB$V_OPENED bit.
!-
    BAS$$CB_PUSH (.CHAN, LUB$K_LUN_MIN);
    SAVE_CCB = .CCB;

    IF ( NOT .CCB [LUB$V_OPENED]) THEN BAS$$STOP (BAS$K_VIRARROPE);

!+
! If the channel was not opened with organization VIRTUAL, reject it.  This
! also catches channel 0, which is always open but never has VIRTUAL
! organization.
!-

    IF (.CCB [LUB$B_ORGAN] NEQ LUB$K_ORG_VIRTU) THEN BAS$$STOP_IO (BAS$K_VIRARRDIS);

!+
! If this channel has been used for block I/O, reject it.
!-

    IF (.CCB [LUB$V_BLK_USE]) THEN BAS$$STOP_IO (BAS$K_ILLOPE);

!+
! If the recordsize of the file is not 512 bytes, reject it.
!-

    IF (.CCB [RAB$W_USZ] NEQ K_BLOCK_LENGTH) THEN BAS$$STOP_IO (BAS$K_VIRBUFTOO);

!+
! If the file is marked read only, reject it.
!-

    IF (.CCB [LUB$V_READ_ONLY]) THEN BAS$$STOP_IO (BAS$K_ILLILLACC);

!+
! Mark the LUB as being used for a virtual array.
!-
    CCB [LUB$V_VA_USE] = 1;
!+
! Record access will always be by key
!-
    CCB [RAB$B_RAC] = RAB$C_KEY;
!+
2 ! Set the RAB so that a $GET to a non-existent record will still lock
2 ! that record.
2 !-
2    CCB [RAB$V_NXR] = 1;
2 !+
2 ! Set the address of our CLOSE appendage in the LUB.  If somebody else's
2 ! is already there, we have a serious problem.
2 !-
2
2    IF (.CCB [LUB$A_CLOSE] EQLA 0) THEN CCB [LUB$A_CLOSE] = BAS$$VA_CLOSE;
2
2    IF (.CCB [LUB$A_CLOSE] NEQA BAS$$VA_CLOSE) THEN BAS$$STOP_IO (BAS$K_PROLOSSOR);
2
2 !+
2 ! If this is not the first reference to this file, we may have to
2 ! write out the current buffer.  We will write only if the current buffer
2 ! is not the buffer we wish to access.  LUB$L_LOG_RECNO is initialized
2 ! to zero for virtual files.
2 !-
2
3    IF (.CCB [LUB$L_LOG_RECNO] EQL ((.INDEX + .DESCRIP [DSC$L_BYTEOFF])/K_BLOCK_LENGTH) + 1)
2    THEN
2        READ_RECORD = 0
2    ELSE
3        BEGIN
3 !+
3 ! We actually do the PUT only if the buffer has been changed since we last
3 ! read it, as recorded by LUB$V_OUTBUF_DR.
3 !-
3
4        IF (.CCB [LUB$V_OUTBUF_DR])
3        THEN
4            BEGIN
4            PUT_STATUS = $PUT (RAB = .CCB);
4
4        IF .PUT_STATUS EQL RMS$_CONTROLC
4        THEN
4            BAS$$SIGNAL_CTRLC ();
4
4 !+
4 ! If the PUT fails, we must worry about the RSA error, which can happen if
4 ! we are running at AST level, and the AST interrupted some RMS I/O.  If
4 ! we get this error, wait for it to go away.  Any other RMS error is fatal.
4 !-
4
5            IF ( NOT .PUT_STATUS)
4            THEN
5                BEGIN
5
5                WHILE (.PUT_STATUS EQL RMS$_RSA) DO
6                    BEGIN
6                    $WAIT (RAB = .CCB);
6                    PUT_STATUS = $PUT (RAB = .CCB);
6
6                    IF .PUT_STATUS EQL RMS$_CONTROLC
6                    THEN
6                        BAS$$SIGNAL_CTRLC ();
```

```
                    END;

                IF ( NOT .PUT_STATUS) THEN BAS$$STOP_IO (BAS$K_IOERR_REC);

                END;

!+
! The buffer is no longer "dirty", mark it so.
!-

            CCB [LUB$V_OUTBUF_DR] = 0;
            END;

        READ_RECORD = 1;
        END;

!+
! If necessary, read in the record containing the element we want.
!-

    IF (.READ_RECORD)
    THEN
        BEGIN
        CCB [LUB$L_LOG_RECNO] = ((.INDEX + .DESCRIP [DSC$L_BYTEOFF])/K_BLOCK_LENGTH) + 1;
        GET_STATUS = $GET (RAB = .CCB);

        IF .GET_STATUS EQL RMS$_CONTROLC
        THEN
            BAS$$SIGNAL_CTRLC ();

!+
! If we are at EOF, extend the file.  This is compatable with the PDP-11.
!-

        IF ((.GET_STATUS EQL RMS$_EOF) OR (.GET_STATUS EQL RMS$_OK_RNF))
        THEN
            BEGIN

            LOCAL
                FAB_BLOCK : $FAB_DECL,
                EXTEND_STATUS;

!+
! If the file is already allocated beyond the current end-of-file point
! (which can happen on disk if the cluster size is greater than 1) then
! do not do any allocation.
!-

            IF (.CCB [LUB$L_LOG_RECNO] GTR .CCB [LUB$L_REC_MAX])
            THEN
                BEGIN
                $FAB_INIT (FAB = FAB_BLOCK);
                FAB_BLOCK [FAB$L_ALQ] = .CCB [LUB$L_LOG_RECNO] - .CCB [LUB$L_REC_MAX];
                FAB_BLOCK [FAB$W_IFI] = .CCB [LUB$W_IFI];
                CCB [LUB$A_FAB] = FAB_BLOCK;
                CCB [RAB$L_STS] = SS$_NORMAL;
                EXTEND_STATUS = $EXTEND (FAB = FAB_BLOCK);
```

```
5                   IF ( NOT .EXTEND_STATUS) THEN BAS$$STOP_IO (BAS$K_IOERR_REC);
5
5                   CCB [LUB$L_REC_MAX] = .CCB [LUB$L_REC_MAX] + .FAB_BLOCK [FAB$L_ALQ];
5                   CCB [LUB$A_FAB] = 0;
4                   END;
4
4   !+
4   ! Since we did not really read a record, make sure the buffer contains
4   ! all zeros.
4   !-
4               CH$FILL (0, .CCB [RAB$W_USZ], .CCB [RAB$L_UBF]);
4               CCB [RAB$W_RSZ] = .CCB [RAB$W_USZ];
4               CCB [RAB$L_RBF] = .CCB [RAB$L_UBF];
4               END
3           ELSE
4               BEGIN
4
5               IF ( NOT .GET_STATUS)
4               THEN
5                   BEGIN
5   !+
5   ! Again, worry about the RSA RMS error.
5   !-
5
5                   WHILE (.GET_STATUS EQL RMS$_RSA) DO
6                       BEGIN
6                       $WAIT (RAB = .CCB);
6                       GET_STATUS = $GET (RAB = .CCB);
6
6                       IF .GET_STATUS EQL RMS$_CONTROLC
6                       THEN
6                           BAS$$SIGNAL_CTRLC ();
6
5                       END;
5
5                   IF ( NOT .GET_STATUS) THEN BAS$$STOP_IO (BAS$K_IOERR_REC);
5
4                   END;
4
3               END;
3
2           END;
2
2   !+
2   ! At this point, the proper record is in the buffer, and we can copy
2   ! data to it.
2   !-
2       IF .DESCRIP [DSC$B_DTYPE] NEQ DSC$K_DTYPE_P
2       THEN
2       CH$MOVE (.DESCRIP [DSC$W_LENGTH], .VALUE,
2           .CCB [RAB$L_RBF] + ((.INDEX + .DESCRIP [DSC$L_BYTEOFF]) MOD K_BLOCK_LENGTH))
2       ELSE
3           BEGIN
3           MAP
3               DESCRIP : REF BLOCK [12,BYTE],
3               VALUE : REF BLOCK [12,BYTE];
```

```
;  739          1449  3          LOCAL
;  740          1450  3              COUNT;
;  741          1451  3          COUNT = .VALUE [DSC$B_SCALE] - .DESCRIP [DSC$B_SCALE];
;  742          1452  3          ASHP (COUNT, VALUE [DSC$W_LENGTH],
;  743          1453  3              .VALUE [DSC$A_POINTER], %REF(0), DESCRIP [DSC$W_LENGTH],
;  744          1454  3              .CCB [RAB$L_RBF] + ((.INDEX + .DESCRIP [DSC$L_BYTEOFF]) MOD K_BLOCK_LENGTH));
;  745          1455  2          END;
;  746          1456  2  !+
;  747          1457  2  ! Since the buffer differs from the disk, mark it "dirty" so it will be
;  748          1458  2  ! written out.
;  749          1459  2  !-
;  750          1460  2      CCB [LUB$V_OUTBUF_DR] = 1;
;  751          1461  2  !+
;  752          1462  2  ! Done with this I/O channel.
;  753          1463  2  !-
;  754          1464  2      BAS$$CB_POP ();
;  755          1465  1      END;                                            ! end of BAS$$VA_STORE



                                                                          .EXTRN   SYS$EXTEND

                                               0FFC 00000                 .ENTRY   BAS$$VA_STORE, Save R2,R3,R4,R5,R6,R7,R8,-  ; 1164
                                                                                   R9,R10,R11                                  ;
                        5A 00000000G   00   9E 00002                      MOVAB    BAS$$STOP_IO, R10                           ;
                        5E        AC   AE   9E 00009                      MOVAB    -84(SP), SP                                 ;
                                  50   AE   D4 0000D                      CLRL     SAVE_CCB                                    ; 1203
                        6D      0222   CF   DE 00010                      MOVAL    26$, (FP)                                   ;
                        57        04   AC   D0 00015                      MOVL     DESCRIP, R7                                 ; 1232
                        52        FC   A7   D0 00019                      MOVL     -4(R7), CHAN                                ;
                                  50   D4   0001D                         CLRL     R0                                          ; 1238
                            00000000G  00   16 0001F                      JSB      BAS$$CB_PUSH                                ;
                  50    AE             5B   D0 00025                      MOVL     CCB, SAVE_CCB                               ; 1239
                        0B        FC   AB   E8 00029                      BLBS     -4(CCB), 1$                                 ; 1241
                        7E       00G   8F   9A 0002D                      MOVZBL   #BAS$K_VIRARROPE, -(SP)                     ;
           00000000G    00             01   FB 00031                      CALLS    #1, BAS$$STOP                               ;
                        05        C4   AB   91 00038  1$:                 CMPB     -60(CCB), #5                                ; 1249
                                  07   13   0003C                         BEQL     2$                                          ;
                        7E       00G   8F   9A 0003E                      MOVZBL   #BAS$K_VIRARRDIS, -(SP)                     ;
                        6A             01   FB 00042                      CALLS    #1, BAS$$STOP_IO                            ;
                        58        FE   AB   9E 00045  2$:                 MOVAB    -2(CCB), R8                                 ; 1255
             07         68             09   E1 00049                      BBC      #9, (R8), 3$                                ;
                        7E       00G   8F   9A 0004D                      MOVZBL   #BAS$K_ILLOPE, -(SP)                        ;
                        6A             01   FB 00051                      CALLS    #1, BAS$$STOP_IO                            ;
                0200    8F        20   AB   B1 00054  3$:                 CMPW     32(CCB), #512                               ; 1261
                                  07   13   0005A                         BEQL     4$                                          ;
                        7E       00G   8F   9A 0005C                      MOVZBL   #BAS$K_VIRBUFTOO, -(SP)                     ;
                        6A             01   FB 00060                      CALLS    #1, BAS$$STOP_IO                            ;
             07    FC   AB             02   E1 00063  4$:                 BBC      #2, -4(CCB), 5$                             ; 1267
                        7E       00G   8F   9A 00068                      MOVZBL   #BAS$K_ILLILLACC, -(SP)                     ;
                        6A             01   FB 0006C                      CALLS    #1, BAS$$STOP_IO                            ;
                  01    A8             01   88 0006F  5$:                 BISB2    #1, 1(R8)                                   ; 1272
                  1E    AB             01   90 00073                      MOVB     #1, 30(CCB)                                 ; 1276
                  06    AB        80   8F   88 00077                      BISB2    #128, 6(CCB)                                ; 1281
                                  A4   AB   D5 0007C                      TSTL     -92(CCB)                                    ; 1287
                                  06   12   0007F                         BNEQ     6$                                          ;
                  A4    AB     0000V   CF   9E 00081                      MOVAB    BAS$$VA_CLOSE, -92(CCB)                     ;
```

```
                                                            C 13
BAS$$VIRT_IO                                                16-Sep-1984 01:28:00    VAX-11 Bliss-32 V4.0-742              Page  19
1-027                                                       14-Sep-1984 11:56:46    [BASRTL.SRC]BASVIRTIO.B32;1                (4)

                             50     0000V  CF  9E 00087 6$:    MOVAB   BAS$$VA_CLOSE, R0                        ; 1289
                             50        A4  AB  D1 0008C        CMPL    -92(CCB), R0                             ;
                                           07  13 00090        BEQL    7$                                       ;
                             7E       00G  8F  9A 00092        MOVZBL  #BAS$K_PROLOSSOR, -(SP)                  ;
                             6A            01  FB 00096        CALLS   #1, BAS$$STOP_IO                         ;
         59        08        AC        F8  A7  C1 00099 7$:    ADDL3   -8(R7), INDEX, R9                        ; 1298
         50                  59 00000200   8F  C7 0009F        DIVL3   #512, R9, R0                             ;
                             53        01  A0  9E 000A7        MOVAB   1(R0), R3                                ;
                             53        E0  AB  D1 000AB        CMPL    -32(CCB), R3                             ;
                                           04  12 000AF        BNEQ    8$                                       ;
                                           54  D4 000B1        CLRL    READ_RECORD                              ; 1300
                                           62  11 000B3        BRB     14$                                      ;
         5B                  68            03  E1 000B5 8$:    BBC     #3, (R8), 13$                            ; 1308
                                           5B  DD 000B9        PUSHL   CCB                                      ; 1311
                  00000000G  00            01  FB 000BB        CALLS   #1, SYS$PUT                              ;
                             52            50  D0 000C2        MOVL    R0, PUT_STATUS                           ;
                  00010651   8F            52  D1 000C5        CMPL    PUT_STATUS, #67153                       ; 1313
                                           07  12 000CC        BNEQ    9$                                       ;
                  00000000G  00            00  FB 000CE        CALLS   #0, BAS$$SIGNAL_CTRLC                    ; 1315
                             39            52  E8 000D5 9$:    BLBS    PUT_STATUS, 12$                          ; 1323
                  000182CA   8F            52  D1 000D8 10$:   CMPL    PUT_STATUS, #99034                       ; 1327
                                           27  12 000DF        BNEQ    11$                                      ;
                                           5B  DD 000E1        PUSHL   CCB                                      ; 1329
                  00000000G  00            01  FB 000E3        CALLS   #1, SYS$WAIT                             ;
                                           5B  DD 000EA        PUSHL   CCB                                      ; 1330
                  00000000G  00            01  FB 000EC        CALLS   #1, SYS$PUT                              ;
                             52            50  D0 000F3        MOVL    R0, PUT_STATUS                           ;
                  00010651   8F            52  D1 000F6        CMPL    PUT_STATUS, #67153                       ; 1332
                                           D9  12 000FD        BNEQ    10$                                      ;
                  00000000G  00            00  FB 000FF        CALLS   #0, BAS$$SIGNAL_CTRLC                    ; 1334
                                           D0  11 00106        BRB     10$                                      ; 1327
                             06            52  E8 00108 11$:   BLBS    PUT_STATUS, 12$                          ; 1338
                             7E            01  CE 0010B        MNEGL   #1, -(SP)                                ;
                             6A            01  FB 0010E        CALLS   #1, BAS$$STOP_IO                         ;
                             68            08  8A 00111 12$:   BICB2   #8, (R8)                                 ; 1345
                             54            01  D0 00114 13$:   MOVL    #1, READ_RECORD                          ; 1348
                             03            54  E8 00117 14$:   BLBS    READ_RECORD, 15$                         ; 1355
                                     00CC  31 0011A            BRW     23$                                      ;
                         E0  AB            53  D0 0011D 15$:   MOVL    R3, -32(CCB)                             ; 1358
                                           5B  DD 00121        PUSHL   CCB                                      ; 1359
                  00000000G  00            01  FB 00123        CALLS   #1, SYS$GET                              ;
                             56            50  D0 0012A        MOVL    R0, GET_STATUS                           ;
                  00010651   8F            56  D1 0012D        CMPL    GET_STATUS, #67153                       ; 1361
                                           07  12 00134        BNEQ    16$                                      ;
                  00000000G  00            00  FB 00136        CALLS   #0, BAS$$SIGNAL_CTRLC                    ; 1363
                  0001827A   8F            56  D1 0013D 16$:   CMPL    GET_STATUS, #98938                       ; 1369
                                           09  13 00144        BEQL    17$                                      ;
                  00018049   8F            56  D1 00146        CMPL    GET_STATUS, #98377                       ;
                                           5E  12 0014D        BNEQ    20$                                      ;
                         E4  AB        E0  AB  D1 0014F 17$:   CMPL    -32(CCB), -28(CCB)                       ; 1383
                                           43  15 00154        BLEQ    19$                                      ;
   0050    8F            00                6E  00  2C 00156    MOVC5   #0, (SP), #0, #80, $RMS_PTR              ; 1386
                                           6E     0015D                                                         ;
                             6E      5003  8F  B0 0015E        MOVW    #20483, $RMS_PTR                         ;
                             16        AE  02  90 00163        MOVB    #2, $RMS_PTR+22                          ;
                             1F        AE  02  90 00167        MOVB    #2, $RMS_PTR+31                          ;
                10 AE        E0        AB  E4  AB  C3 0016B    SUBL3   -28(CCB), -32(CCB), FAB_BLOCK+16         ; 1387
```

```
                    02  AE          D0  AB  B0 00172           MOVW    -48(CCB), FAB_BLOCK+2                        ; 1388
                    E8  AB              6E  9E 00177           MOVAB   FAB_BLOCK, -24(CCB)                          ; 1389
                    08  AB              01  D0 0017B           MOVL    #1, 8(CCB)                                   ; 1390
                                        5E  DD 0017F           PUSHL   SP                                           ; 1391
            00000000G   00              01  FB 00181           CALLS   #1, SYS$EXTEND                               ;
                        06              50  E8 00188           BLBS    EXTEND_STATUS, 18$                           ; 1393
                        7E              01  CE 0018B           MNEGL   #1, -(SP)                                    ;
                        6A              01  FB 0018E           CALLS   #1, BAS$$STOP_IO                             ;
                    E4  AB          10  AE  C0 00191  18$:     ADDL2   FAB_BLOCK+16, -28(CCB)                       ; 1395
                                    E8  AB  D4 00196           CLRL    -24(CCB)                                     ; 1396
    20   AB         00                  6E  00  2C 00199  19$:     MOVC5   #0, (SP), #0, 32(CCB), @36(CCB)          ; 1403
                                    24  BB     0019F                                                            ;
                    22  AB          20  AB  B0 001A1           MOVW    32(CCB), 34(CCB)                             ; 1404
                    28  AB          24  AB  D0 001A6           MOVL    36(CCB), 40(CCB)                             ; 1405
                                        3C  11 001AB           BRB     23$                                          ; 1369
                        39              56  E8 001AD  20$:     BLBS    GET_STATUS, 23$                              ; 1410
            000182DA    8F              56  D1 001B0  21$:     CMPL    GET_STATUS, #99034                           ; 1417
                                        27  12 001B7           BNEQ    22$                                          ;
                                        5B  DD 001B9           PUSHL   CCB                                          ; 1419
            00000000G   00              01  FB 001BB           CALLS   #1, SYS$WAIT                                 ;
                                        5B  DD 001C2           PUSHL   CCB                                          ; 1420
            00000000G   00              01  FB 001C4           CALLS   #1, SYS$GET                                  ;
                        56              50  D0 001CB           MOVL    R0, GET_STATUS                               ;
            00010651    8F              56  D1 001CE           CMPL    GET_STATUS, #67153                           ; 1422
                                        D9  12 001D5           BNEQ    21$                                          ;
            00000000G   00              00  FB 001D7           CALLS   #0, BAS$$SIGNAL_CTRLC                        ; 1424
                                        D0  11 001DE           BRB     21$                                          ; 1417
                        06              56  E8 001E0  22$:     BLBS    GET_STATUS, 23$                              ; 1428
                        7E              01  CE 001E3           MNEGL   #1, -(SP)                                    ;
                        6A              01  FB 001E6           CALLS   #1, BAS$$STOP_IO                             ;
                        56          0C  AC  D0 001E9  23$:     MOVL    VALUE, R6                                    ; 1442
                        15          02  A7  91 001ED           CMPB    2(R7), #21                                   ; 1440
                                        16  13 001F1           BEQL    24$                                          ;
    7E              00                  59  01  7A 001F3           EMUL    #1, R9, #0, -(SP)                        ; 1443
    50              50                  8E 00000200  8F  7B 001F8  EDIV    #512, (SP)+, R0, R0                      ;
               28 BB40                  66  67  28 00201           MOVC3   (R7), (R6), @40(CCB)[R0]                 ;
                                        23  11 00207           BRB     25$                                          ; 1442
                        51          08  A6  98 00209  24$:     CVTBL   8(R6), COUNT                                 ; 1451
                        50          08  A7  98 0020D           CVTBL   8(R7), R0                                    ;
                        51              50  C2 00211           SUBL2   R0, COUNT                                    ;
    7E              00                  59  01  7A 00214           EMUL    #1, R9, #0, -(SP)                        ; 1454
    50              50                  8E 00000200  8F  7B 00219  EDIV    #512, (SP)+, R0, R0                      ;
    00         04   B6                  66  51  F8 00222           ASHP    COUNT, (R6), @4(R6), #0, (R7), @40(CCB)[R0] ;
                  28 BB40               67     00228                                                            ;
                        68              08  88 0022C  25$:     BISB2   #8, (R8)                                     ; 1460
                     00000000G          00  16 0022F           JSB     BAS$$CB_POP                                  ; 1464
                                        04 00235               RET                                              ; 1465
                                      0000 00236      26$:     .WORD   Save nothing                                 ; 1203
                        50          08  AC  D0 00238           MOVL    8(AP), R0                                    ;
                        50          04  A0  D0 0023C           MOVL    4(R0), R0                                    ;
                                    FC  A0  9F 00240           PUSHAB  SAVE_CCB                                     ;
                                        01  DD 00243           PUSHL   #1                                           ;
                                        5E  DD 00245           PUSHL   SP                                           ;
                        7E          04  AC  7D 00247           MOVQ    4(AP), -(SP)                                 ;
                0000V   CF              03  FB 0024B           CALLS   #3, HANDLER                                  ;
                                        04 00250               RET                                              ;
```

```
; Routine Size:  593 bytes,    Routine Base:  _BAS$CODE + 01E5
```

```
1 GLOBAL ROUTINE BAS$$WHOLE_VA_FETCH (                    ! Fetch routine
1         DESCRIP,                                        ! The descriptor for this virtual array
1         VALUE                                           ! Where to store array item
1     ) : NOVALUE =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Fetch all values from a virtual array.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       DESCRIP.mz.r    The descriptor for the virtual array
1 !       VALUE.wz.r      Address of the 1st location to store
1 !                       the values.  The number of bytes
1 !                       to store is in the LENGTH field of
1 !                       DESCRIP.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !
1 !       Signals if an error is encountered.
1 !
1 !--
1
2     BEGIN
2
2     MAP
2         DESCRIP : REF BLOCK [8, BYTE];
2
2     GLOBAL REGISTER
2         CCB = K_CCB_REG : REF BLOCK [, BYTE];
2
2     LOCAL
2         CHAN,                                           ! The channel this array is defined on
2         HANDLE,                                         ! Pointer to info for this array
2         GET_STATUS,                                     ! Last RMS GET status
2         DEST,                                           ! Updated VALUE
2         LEN,                                            ! Number of bytes to move
2         REMAINING_BYTES,                                ! Number of bytes left to move
2         SAVE_CCB : VOLATILE,                            ! CCB to POP, or zero.
2         PUT_STATUS;                                     ! status retd by RMS $PUT
2
2 !+
2 ! Establish a handler to pop the CCB when unwinding.
```

```
2 !-
2
2       ENABLE
2           HANDLER (SAVE_CCB);
2
2 !+
2 ! DEST is initially the same as VALUE, but will be updated as blocks
2 ! of 512 bytes are moved.
2 !-
2
2       DEST = .VALUE;
2
2 !+
2 ! Block #1 may not be entirely this array.  Initialize REMAINING_BYTES
2 ! to 512 minus whatever offset there is.
2 !-
2
2       IF (.DESCRIP [DSC$L_ARSIZE] + .DESCRIP [DSC$L_BYTEOFF]) LSS K_BLOCK_LENGTH
2       THEN
2           REMAINING_BYTES = .DESCRIP [DSC$L_ARSIZE]
2       ELSE
2           REMAINING_BYTES = K_BLOCK_LENGTH - .DESCRIP [DSC$L_BYTEOFF];
2
2 !+
2 ! Fetch the array's channel number from its descriptor
2 !-
2       CHAN = .DESCRIP [DSC$L_LOGUNIT];
2 !+
2 ! Get a pointer to the LUB/ISB/RAB for this channel.  If the channel has not
2 ! been opened yet, this call will allocate the LUB/ISB/RAB, but we will reject
2 ! it for lack of the LUB$V_OPENED bit.
2 !-
2       BAS$$CB_PUSH (.CHAN, LUB$K_LUN_MIN);
2       SAVE_CCB = .CCB;
2
2       IF ( NOT .CCB [LUB$V_OPENED]) THEN BAS$$STOP (BAS$K_VIRARROPE);
2
2 !+
2 ! If the channel was not opened with organization VIRTUAL, reject it.  This
2 ! also catches channel 0, which is always open but never has VIRTUAL
2 ! organization.
2 !-
2
2       IF (.CCB [LUB$B_ORGAN] NEQ LUB$K_ORG_VIRTU) THEN BAS$$STOP_IO (BAS$K_VIRARRDIS);
2
2 !+
2 ! If this channel has been used for block I/O, reject it.
2 !-
2
2       IF (.CCB [LUB$V_BLK_USE]) THEN BAS$$STOP_IO (BAS$K_ILLOPE);
2
2 !+
2 ! If the record size declared for the file is not 512 bytes, reject it.
2 !-
2
2       IF (.CCB [RAB$W_USZ] NEQ K_BLOCK_LENGTH) THEN BAS$$STOP_IO (BAS$K_VIRBUFTOO);
2
```

```
2 !+
2 ! Mark the LUB as being used for a virtual array.
2 !-
2     CCB [LUB$V_VA_USE] = 1;
2 !+
2 ! Record access will always be by key
2 !-
2     CCB [RAB$B_RAC] = RAB$C_KEY;
2 !+
2 ! Mark the RAB so that a $GET to a non-existent record will still lock it.
2 !-
2     CCB [RAB$V_NXR] = 1;
2 !+
2 ! Set the address of our CLOSE appendage in the LUB.  If somebody else's
2 ! is already there, we have a serious problem.
2 !-
2
2     IF (.CCB [LUB$A_CLOSE] EQLA 0) THEN CCB [LUB$A_CLOSE] = BAS$$VA_CLOSE;
2
2     IF (.CCB [LUB$A_CLOSE] NEQA BAS$$VA_CLOSE) THEN BAS$$STOP_IO (BAS$K_PROLOSSOR);
2
2 !+
2 ! If this is not the first reference to the file, we may have to write out
2 ! the current buffer.
2 !-
2
2     IF .CCB [LUB$L_LOG_RECNO] NEQ 0
2     THEN
3         BEGIN
4         IF (.CCB [LUB$V_OUTBUF_DR])      ! buffer has been changed
3         THEN
4             BEGIN
4             PUT_STATUS = $PUT (RAB = .CCB);
4
4             IF .PUT_STATUS EQL RMS$_CONTROLC
4             THEN
4                 BAS$$SIGNAL_CTRLC ();
4
5             IF (NOT .PUT_STATUS)
4             THEN
5                 BEGIN
5                 WHILE (.PUT_STATUS EQL RMS$_RSA) DO
6                     BEGIN
6                     $WAIT (RAB = .CCB);
6                     PUT_STATUS = $PUT (RAB = .CCB);
6
6                     IF .PUT_STATUS EQL RMS$_CONTROLC
6                     THEN
6                         BAS$$SIGNAL_CTRLC ();
6
5                     END;
5                 IF (NOT .PUT_STATUS) THEN BAS$$STOP_IO (BAS$K_IOERR_REC);
4                 END;
4             CCB [LUB$V_OUTBUF_DR] = 0;
3             END;
2         END;
2
```

```
2 !+
2 ! Calculate the number of blocks in the virtual array and loop through
2 ! all of them.
2 !-
2
2       INCR BLKCNT FROM 1 TO ((.DESCRIP [DSC$L_BYTEOFF] + .DESCRIP [DSC$L_ARSIZE])/512 + 1) DO
3           BEGIN
3           CCB [LUB$L_LOG_RECNO] = .BLKCNT;
3           GET_STATUS = $GET (RAB = .CCB);
3
3           IF .GET_STATUS EQL RMS$_CONTROLC
3           THEN
3               BAS$$SIGNAL_CTRLC ();
3
3 !+
3 ! If we get EOF, just clear the buffer.  This is compatable with
3 ! the PDP-11.
3 !-
3
4           IF ((.GET_STATUS EQL RMS$_EOF) OR (.GET_STATUS EQL RMS$_OK_RNF))
3           THEN
4               BEGIN
4               CH$FILL (0, .CCB [RAB$W_USZ], .CCB [RAB$L_UBF]);
4               CCB [RAB$L_RBF] = .CCB [RAB$L_UBF];
4               CCB [RAB$W_RSZ] = .CCB [RAB$W_USZ];
4               END
3           ELSE
4               BEGIN
4
5               IF ( NOT .GET_STATUS)
4               THEN
5                   BEGIN
5 !+
5 ! Again, worry about the RSA RMS error.
5 !-
5
5                   WHILE (.GET_STATUS EQL RMS$_RSA) DO
6                       BEGIN
6                       $WAIT (RAB = .CCB);
6                       GET_STATUS = $GET (RAB = .CCB);
6
6                       IF .GET_STATUS EQL RMS$_CONTROLC
6                       THEN
6                           BAS$$SIGNAL_CTRLC ();
6
5                       END;
5
5                   IF ( NOT .GET_STATUS) THEN BAS$$STOP_IO (BAS$K_IOERR_REC);
5
4                   END;
4
3               END;
3
3 !+
3 ! Copy the 512 byte buffer to the desired location.  If this is the last
3 ! buffer (or the first), there may not be 512 bytes so check for this.
3 !-
```

```
;  985          1694  3              LEN = MIN (K_BLOCK_LENGTH, .REMAINING_BYTES);
;  986          1695  3              CH$MOVE (.LEN, (.CCB [RAB$L_RBF] + (IF .BLKCNT EQL 1
;  987          1696  5                                                   THEN .DESCRIP [DSC$L_BYTEOFF]
;  988          1697  5                                                   ELSE 0)), .DEST);
;  989          1698  3
;  990          1699  3              DEST = .DEST + .LEN;
;  991          1700  3              REMAINING_BYTES = .REMAINING_BYTES - .LEN;
;  992          1701  3
;  993          1702  2              END;                                        ! loop through all blocks in v.a.
;  994          1703  2
;  995          1704  2      !+
;  996          1705  2      ! Done with this I/O channel.
;  997          1706  2      !-
;  998          1707  2          BAS$$CB_POP ();
;  999          1708  1          END;                                            ! end of BAS$$WHOLE_VA_FETCH



                                           0FFC 00000           .ENTRY   BAS$$WHOLE_VA_FETCH, Save R2,R3,R4,R5,R6,-    ; 1466
                                                                         R7,R8,R9,R10,R11                              ;
                    5E                08   C2 00002             SUBL2    #8, SP                                        ;
                             04       AE   D4 00005             CLRL     SAVE_CCB                                      ; 1503
                    6D     01EA       CF   DE 00008             MOVAL    25$, (FP)                                     ;
                             08       AC   DD 0000D             PUSHL    VALUE                                         ; 1533
                    57       04       AC   D0 00010             MOVL     DESCRIP, R7                                   ; 1540
        50          0C       A7    F8 A7   C1 00014             ADDL3    -8(R7), 12(R7), R0                            ;
           00000200 8F                50   D1 0001A             CMPL     R0, #512                                      ;
                                      06   18 00021             BGEQ     1$                                            ;
                    56       0C       A7   D0 00023             MOVL     12(R7), REMAINING_BYTES                       ; 1542
                                      09   11 00027             BRB      2$                                            ;
        56 00000200 8F       F8       A7   C3 00029  1$:        SUBL3    -8(R7), #512, REMAINING_BYTES                 ; 1544
                    52       FC       A7   D0 00032  2$:        MOVL     -4(R7), CHAN                                  ; 1549
                                      50   D4 00036             CLRL     R0                                            ; 1555
                        00000000G     00   16 00038             JSB      BAS$$CB_PUSH                                  ;
                08  AE                5B   D0 0003E             MOVL     CCB, SAVE_CCB                                 ; 1556
                    0B       FC       AB   E8 00042             BLBS     -4(CCB), 3$                                   ; 1558
                    7E      00G       8F   9A 00046             MOVZBL   #BAS$K_VIRARRPOPE, -(SP)                      ;
           00000000G 00               01   FB 0004A             CALLS    #1, BAS$$STOP                                 ;
                    05       C4       AB   91 00051  3$:        CMPB     -60(CCB), #5                                  ; 1566
                                      0B   13 00055             BEQL     4$                                            ;
                    7E      00G       8F   9A 00057             MOVZBL   #BAS$K_VIRARRDIS, -(SP)                       ;
           00000000G 00               01   FB 0005B             CALLS    #1, BAS$$STOP_IO                              ;
        0B          FF AB             01   E1 00062  4$:        BBC      #1, -1(CCB), 5$                               ; 1572
                    7E      00G       8F   9A 00067             MOVZBL   #BAS$K_ILLOPE, -(SP)                          ;
           00000000G 00               01   FB 0006B             CALLS    #1, BAS$$STOP_IO                              ;
               0200 8F       20       AB   B1 00072  5$:        CMPW     32(CCB), #512                                 ; 1578
                                      0B   13 00078             BEQL     6$                                            ;
                    7E      00G       8F   9A 0007A             MOVZBL   #BAS$K_VIRBUFTOO, -(SP)                       ;
           00000000G 00               01   FB 0007E             CALLS    #1, BAS$$STOP_IO                              ;
                 FF AB                01   88 00085  6$:        BISB2    #1, -1(CCB)                                   ; 1583
                 1E AB                01   90 00089             MOVB     #1, 30(CCB)                                   ; 1587
                 06 AB       80       8F   88 0008D             BISB2    #128, 6(CCB)                                  ; 1591
                             A4       AB   D5 00092             TSTL     -92(CCB)                                      ; 1597
                                      06   12 00095             BNEQ     7$                                            ;
                 A4 AB     0000V      CF   9E 00097             MOVAB    BAS$$VA_CLOSE, -92(CCB)                       ;
```

```
                                 50      0000V CF 9E 0009D 7$:   MOVAB   BAS$$VA_CLOSE, R0                          ; 1599
                                 50         A4 AB D1 000A2       CMPL    -92(CCB), R0                               ;
                                               0B 13 000A6       BEQL    8$                                         ;
                                 7E        00G 8F 9A 000A8       MOVZBL  #BAS$K_PROLOSSOR, -(SP)                    ;
                       00000000G 00            01 FB 000AC       CALLS   #1, BAS$$STOP_IO                           ;
                                            E0 AB D5 000B3 8$:   TSTL    -32(CCB)                                   ; 1606
                                               65 13 000B6       BEQL    13$                                        ;
                    60        FE AB            03 E1 000B8       BBC     #3, -2(CCB), 13$                           ; 1609
                                               5B DD 000BD       PUSHL   CCB                                        ; 1612
                       00000000G 00            01 FB 000BF       CALLS   #1, SYS$PUT                                ;
                                 52            50 D0 000C6       MOVL    R0, PUT_STATUS                             ;
                       00010651  8F            52 D1 000C9       CMPL    PUT_STATUS, #67153                         ; 1614
                                               07 12 000D0       BNEQ    9$                                         ;
                       00000000G 00            00 FB 000D2       CALLS   #0, BAS$$SIGNAL_CTRLC                      ; 1616
                                 3D            52 E8 000D9 9$:   BLBS    PUT_STATUS, 12$                            ; 1618
                       000182DA  8F            52 D1 000DC 10$:  CMPL    PUT_STATUS, #99034                         ; 1621
                                               27 12 000E3       BNEQ    11$                                        ;
                                               5B DD 000E5       PUSHL   CCB                                        ; 1623
                       00000000G 00            01 FB 000E7       CALLS   #1, SYS$WAIT                               ;
                                               5B DD 000EE       PUSHL   CCB                                        ; 1624
                       00000000G 00            01 FB 000F0       CALLS   #1, SYS$PUT                                ;
                                 52            50 D0 000F7       MOVL    R0, PUT_STATUS                             ;
                       00010651  8F            52 D1 000FA       CMPL    PUT_STATUS, #67153                         ; 1626
                                               D9 12 00101       BNEQ    10$                                        ;
                       00000000G 00            00 FB 00103       CALLS   #0, BAS$$SIGNAL_CTRLC                      ; 1628
                                               D0 11 0010A       BRB     10$                                        ; 1621
                                 0A            52 E8 0010C 11$:  BLBS    PUT_STATUS, 12$                            ; 1631
                                 7E            01 CE 0010F       MNEGL   #1, -(SP)                                  ;
                       00000000G 00            01 FB 00112       CALLS   #1, BAS$$STOP_IO                           ;
                              FE AB            08 8A 00119 12$:  BICB2   #8, -2(CCB)                                ; 1633
                    50        F8 A7         0C A7 C1 0011D 13$:  ADDL3   12(R7), -8(R7), R0                         ; 1642
                                 50   00000200 8F C6 00123       DIVL2   #512, R0                                   ;
                              04 AE         01 A0 9E 0012A       MOVAB   1(R0), 4(SP)                               ;
                                               59 D4 0012F       CLRL    BLKCNT                                     ; 1696
                                             00B4 31 00131       BRW     24$                                        ;
                              E0 AB            59 D0 00134 14$:  MOVL    BLKCNT, -32(CCB)                           ; 1644
                                               5B DD 00138       PUSHL   CCB                                        ; 1645
                       00000000G 00            01 FB 0013A       CALLS   #1, SYS$GET                                ;
                                 58            50 D0 00141       MOVL    R0, GET_STATUS                             ;
                       00010651  8F            58 D1 00144       CMPL    GET_STATUS, #67153                         ; 1647
                                               07 12 0014B       BNEQ    15$                                        ;
                       00000000G 00            00 FB 0014D       CALLS   #0, BAS$$SIGNAL_CTRLC                      ; 1649
                       0001827A  8F            58 D1 00154 15$:  CMPL    GET_STATUS, #98938                         ; 1656
                                               09 13 0015B       BEQL    16$                                        ;
                       00018049  8F            58 D1 0015D       CMPL    GET_STATUS, #98377                         ;
                                               14 12 00164       BNEQ    17$                                        ;
20   AB             00           6E            00 2C 00166 16$:  MOVC5   #0, (SP), #0, 32(CCB), @36(CCB)            ; 1659
                                            24 BB    0016C                                                          ;
                              28 AB         24 AB D0 0016E       MOVL    36(CCB), 40(CCB)                           ; 1660
                              22 AB         20 AB B0 00173       MOVW    32(CCB), 34(CCB)                           ; 1661
                                               40 11 00178       BRB     20$                                        ; 1656
                                 3D            58 E8 0017A 17$:  BLBS    GET_STATUS, 20$                            ; 1666
                       000182DA  8F            58 D1 0017D 18$:  CMPL    GET_STATUS, #99034                         ; 1673
                                               27 12 00184       BNEQ    19$                                        ;
                                               5B DD 00186       PUSHL   CCB                                        ; 1675
                       00000000G 00            01 FB 00188       CALLS   #1, SYS$WAIT                               ;
                                               5B DD 0018F       PUSHL   CCB                                        ; 1676
```

```
                                                           L 13
BAS$$VIRT_IO                                       16-Sep-1984 01:28:00    VAX-11 Bliss-32 V4.0-742          Page  28
1-027                                              14-Sep-1984 11:56:46    [BASRTL.SRC]BASVIRTIO.B32;1             (5)

                        00000000G  00          01  FB 00191          CALLS    #1, SYS$GET                        :
                                   58          50  D0 00198          MOVL     R0, GET_STATUS                     :
                        00010651   8F          58  D1 0019B          CMPL     GET_STATUS, #67153                 : 1678
                                               D9  12 001A2          BNEQ     18$                                :
                        00000000G  00          00  FB 001A4          CALLS    #0, BAS$$SIGNAL_CTRLC              : 1680
                                               D0  11 001AB          BRB      18$                                : 1673
                                   0A          58  E8 001AD 19$:     BLBS     GET_STATUS, 20$                    : 1684
                                   7E          01  CE 001B0          MNEGL    #1, -(SP)                          :
                        00000000G  00          01  FB 001B3          CALLS    #1, BAS$$STOP_IO                   :
                                   50          56  D0 001BA 20$:     MOVL     REMAINING_BYTES, R0                : 1695
                        00000200   8F          50  D1 001BD          CMPL     R0, #512                           :
                                   05          15  001C4             BLEQ     21$                                :
                                   50     0200 8F  3C 001C6          MOVZWL   #512, R0                           :
                                   5A          50  D0 001CB 21$:     MOVL     R0, LEN                            :
                                   01          59  D1 001CE          CMPL     BLKCNT, #1                         : 1696
                                               06  12 001D1          BNEQ     22$                                :
                                   50       F8 A7  D0 001D3          MOVL     -8(R7), R0                         : 1697
                                               02  11 001D7          BRB      23$                                :
                                               50  D4 001D9 22$:     CLRL     R0                                 : 1696
                 00    BE       28 BB40        5A  28 001DB 23$:     MOVC3    LEN, @40(CCB)[R0], @DEST           : 1698
                                   6E          5A  C0 001E2          ADDL2    LEN, DEST                          : 1699
                                   56          5A  C2 001E5          SUBL2    LEN, REMAINING_BYTES               : 1700
         FF45          59          01       04 AE  F1 001E8 24$:     ACBL     4(SP), #1, BLKCNT, 14$             : 1642
                                      00000000G 00 16 001EF          JSB      BAS$$CB_POP                        : 1707
                                               04  001F5             RET                                         : 1708
                                             0000  001F6 25$:        .WORD    Save nothing                       : 1503
                                   50       08 AC  D0 001F8          MOVL     8(AP), R0                          :
                                   50       04 A0  D0 001FC          MOVL     4(R0), R0                          :
                                            FC A0  9F 00200          PUSHAB   SAVE_CCB                           :
                                               01  DD 00203          PUSHL    #1                                 :
                                               5E  DD 00205          PUSHL    SP                                 :
                                   7E       04 AC  7D 00207          MOVQ     4(AP), -(SP)                       :
                           0000V   CF          03  FB 0020B          CALLS    #3, HANDLER                        :
                                               04  00210             RET                                         :

; Routine Size:  529 bytes,    Routine Base:  _BAS$CODE + 0436


; 1000          1709  1
```

```
1 GLOBAL ROUTINE BAS$$WHOLE_VA_STORE (                      ! Store routine
1         DESCRIP,                                          ! The descriptor for this virtual array
1         VALUE                                             ! Where to find items for array
1     ) : NOVALUE =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Store a values in a virtual array.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       DESCRIP.mz.r    The descriptor for the virtual array
1 !       VALUE.rz.r      The place from which to fetch the value.  The
1 !                       number of bytes to store is in the LENGTH field
1 !                       of DESCRIP.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !
1 !       Signals if an error is encountered.
1 !
1 !--
1
2     BEGIN
2
2     MAP
2         DESCRIP : REF BLOCK [8, BYTE];
2
2     GLOBAL REGISTER
2         CCB = K_CCB_REG : REF BLOCK [, BYTE];
2
2     LOCAL
2         CHAN,                                             ! The channel this array is defined on
2         HANDLE,                                           ! Pointer to info for this array
2         GET_STATUS,                                       ! Last RMS GET status
2         PUT_STATUS,                                       ! Last RMS PUT status
2         SOURCE,                                           ! Address of value
2         LEN,                                              ! Length of values to move
2         REMAINING_BYTES,                                  ! Number of bytes left to move
2         SAVE_CCB : VOLATILE;                              ! CCB to POP, or 0
2
2 !+
2 ! Establish a handler to pop the CCB on unwind.
2 !-
```

```

    ENABLE
        HANDLER (SAVE_CCB);

!+
! SOURCE is initially VALUE until some values have been copied to the
! virtual array.
!-

    SOURCE = .VALUE;

!+
! The first buffer may contain data from a previous array.  Subtract the
! offset from the normal 512 bytes.
!-

    IF (.DESCRIP [DSC$L_ARSIZE] + .DESCRIP [DSC$L_BYTEOFF]) LSS K_BLOCK_LENGTH
    THEN
        REMAINING_BYTES = .DESCRIP [DSC$L_ARSIZE]
    ELSE
        REMAINING_BYTES = K_BLOCK_LENGTH - .DESCRIP [DSC$L_BYTEOFF];

!+
! Fetch the array's channel number from its descriptor
!-
    CHAN = .DESCRIP [DSC$L_LOGUNIT];
!+
! Get a pointer to the LUB/ISB/RAB for this channel.  If the channel has not
! been opened yet, this call will allocate the LUB/ISB/RAB, but we will reject
! it for lack of the LUB$V_OPENED bit.
!-
    BAS$$CB_PUSH (.CHAN, LUB$K_LUN_MIN);
    SAVE_CCB = .CCB;

    IF ( NOT .CCB [LUB$V_OPENED]) THEN BAS$$STOP (BAS$K_VIRARROPE);

!+
! If the channel was not opened with organization VIRTUAL, reject it.  This
! also catches channel 0, which is always open but never has VIRTUAL
! organization.
!-

    IF (.CCB [LUB$B_ORGAN] NEQ LUB$K_ORG_VIRTU) THEN BAS$$STOP_IO (BAS$K_VIRARRDIS);

!+
! If this channel has been used for block I/O, reject it.
!-

    IF (.CCB [LUB$V_BLK_USE]) THEN BAS$$STOP_IO (BAS$K_ILLOPE);

!+
! If the recordsize of the file is not 512 bytes, reject it.
!-

    IF (.CCB [RAB$W_USZ] NEQ K_BLOCK_LENGTH) THEN BAS$$STOP_IO (BAS$K_VIRBUFTOO);

!+
2  ! If the file is marked read only, reject it.
2  !-
2
2      IF (.CCB [LUB$V_READ_ONLY]) THEN BAS$$STOP_IO (BAS$K_ILLILLACC);
2
2  !+
2  ! Mark the LUB as being used for a virtual array.
2  !-
2      CCB [LUB$V_VA_USE] = 1;
2  !+
2  ! Record access will always be by key
2  !-
2      CCB [RAB$B_RAC] = RAB$C_KEY;
2  !+
2  ! Set the RAB so that a $GET to a non-existent record will still lock
2  ! that record.
2  !-
2      CCB [RAB$V_NXR] = 1;
2  !+
2  ! Set the address of our CLOSE appendage in the LUB.  If somebody else's
2  ! is already there, we have a serious problem.
2  !-
2
2      IF (.CCB [LUB$A_CLOSE] EQLA 0) THEN CCB [LUB$A_CLOSE] = BAS$$VA_CLOSE;
2
2      IF (.CCB [LUB$A_CLOSE] NEQA BAS$$VA_CLOSE) THEN BAS$$STOP_IO (BAS$K_PROLOSSOR);
2
2  !+
2  ! If this is not the first reference to the file, we may have to write out
2  ! the current buffer.
2  !-
2
3      IF (.CCB [LUB$L_LOG_RECNO] NEQ 0)
2      THEN
3          BEGIN
4          IF (.CCB [LUB$V_OUTBUF_DR])       ! only write if buffer changed
3          THEN
4              BEGIN
4              PUT_STATUS = $PUT (RAB = .CCB);
4
4              IF .PUT_STATUS EQL RMS$_CONTROLC
4              THEN
4                  BAS$$SIGNAL_CTRLC ();
4
5              IF (NOT .PUT_STATUS)
4              THEN
5                  BEGIN
5                  WHILE (.PUT_STATUS EQL RMS$_RSA) DO
6                      BEGIN
6                      $WAIT (RAB = .CCB);
6                      PUT_STATUS = $PUT (RAB = .CCB);
6
6                      IF .PUT_STATUS EQL RMS$_CONTROLC
6                      THEN
6                          BAS$$SIGNAL_CTRLC ();
6
5                      END;
```

```
5                       IF (NOT .PUT_STATUS) THEN BAS$$STOP_IO (BAS$K_IOERR_REC);
4                       END;
4               CCB [LUB$V_OUTBUF_DR] = 0;
3               END;
2           END;
2
2   !+
2   ! Loop through all the values to be stored.  Moves are done in blocks of 512
2   ! bytes, except for possibly the last buffer which may not be full.
2   !-
2
2       INCR BLKCNT FROM 1 TO ((.DESCRIP [DSC$L_BYTEOFF] + .DESCRIP [DSC$L_ARSIZE])/512 + 1) DO
3           BEGIN
3           CCB [LUB$L_LOG_RECNO] = .BLKCNT;
3           GET_STATUS = $GET (RAB = .CCB);
3
3           IF .GET_STATUS EQL RMS$_CONTROLC
3           THEN
3               BAS$$SIGNAL_CTRLC ();
3
3   !+
3   ! If we are at EOF, extend the file.  This is compatable with the PDP-11.
3   !-
3
4           IF ((.GET_STATUS EQL RMS$_EOF) OR (.GET_STATUS EQL RMS$_OK_RNF))
3           THEN
4               BEGIN
4
4               LOCAL
4                   FAB_BLOCK : $FAB_DECL,
4                   EXTEND_STATUS;
4
4   !+
4   ! If the file is already allocated beyond the current end-of-file point
4   ! (which can happen on disk if the cluster size is greater than 1) then
4   ! do not do any allocation.
4   !-
4
5               IF (.CCB [LUB$L_LOG_RECNO] GTR .CCB [LUB$L_REC_MAX])
4               THEN
5                   BEGIN
5                   $FAB_INIT (FAB = FAB_BLOCK);
5                   FAB_BLOCK [FAB$L_ALQ] = .CCB [LUB$L_LOG_RECNO] - .CCB [LUB$L_REC_MAX];
5                   FAB_BLOCK [FAB$W_IFI] = .CCB [LUB$W_IFI];
5                   CCB [LUB$A_FAB] = FAB_BLOCK;
5                   CCB [RAB$L_STS] = SS$_NORMAL;
5                   EXTEND_STATUS = $EXTEND (FAB = FAB_BLOCK);
5
5                   IF ( NOT .EXTEND_STATUS) THEN BAS$$STOP_IO (BAS$K_IOERR_REC);
5
5                   CCB [LUB$L_REC_MAX] = .CCB [LUB$L_REC_MAX] + .FAB_BLOCK [FAB$L_ALQ];
5                   CCB [LUB$A_FAB] = 0;
4                   END;
4
4   !+
4   ! Since we did not really read a record, make sure the buffer contains
4   ! all zeros.
```

```
4 !-
4               CH$FILL (0, .CCB [RAB$W_USZ], .CCB [RAB$L_UBF]);
4               CCB [RAB$W_RSZ] = .CCB [RAB$W_USZ];
4               CCB [RAB$L_RBF] = .CCB [RAB$L_UBF];
4               END
3           ELSE
4               BEGIN
4
5               IF ( NOT .GET_STATUS)
4               THEN
5                   BEGIN
5 !+
5 ! Again, worry about the RSA RMS error.
5 !-
5
5                   WHILE (.GET_STATUS EQL RMS$_RSA) DO
6                       BEGIN
6                       $WAIT (RAB = .CCB);
6                       GET_STATUS = $GET (RAB = .CCB);
6
6                       IF .GET_STATUS EQL RMS$_CONTROLC
6                       THEN
6                           BAS$$SIGNAL_CTRLC ();
6
5                       END;
5
5                   IF ( NOT .GET_STATUS) THEN BAS$$STOP_IO (BAS$K_IOERR_REC);
5
4                   END;
4
3               END;
3
3 !+
3 ! Move data from the value address to the virtual array.
3 !-
3
3       LEN = MIN (K_BLOCK_LENGTH, .REMAINING_BYTES);
4       CH$MOVE (.LEN, .SOURCE, (.CCB [RAB$L_RBF] +
3           (IF .BLKCNT EQL 1 THEN .DESCRIP [DSC$L_BYTEOFF] ELSE 0)));
3
3 !+
3 ! Write out the buffer.
3 !-
3
3       PUT_STATUS = $PUT (RAB = .CCB);
3
3       IF .PUT_STATUS EQL RMS$_CONTROLC
3       THEN
3           BAS$$SIGNAL_CTRLC ();
3
3 !+
3 ! If the PUT fails, we must worry about the RSA error, which can happen if
3 ! we are running at AST level, and the AST interrupted some RMS I/O.  If
3 ! we get this error, wait for it to go away.  Any other RMS error is fatal.
3 !-
3
4       IF (NOT .PUT_STATUS)
```

```
; 1287          1995  3      THEN
; 1288          1996  4          BEGIN
; 1289          1997  4          WHILE (.PUT_STATUS EQL RMS$_RSA) DO
; 1290          1998  5              BEGIN
; 1291          1999  5              $WAIT (RAB = .CCB);
; 1292          2000  5              PUT_STATUS = $PUT (RAB = .CCB);
; 1293          2001  5
; 1294          2002  5              IF .PUT_STATUS EQL RMS$_CONTROLC
; 1295          2003  5              THEN
; 1296          2004  5                  BAS$$SIGNAL_CTRLC ();
; 1297          2005  5
; 1298          2006  4              END;
; 1299          2007  4
; 1300          2008  4          IF (NOT .PUT_STATUS) THEN BAS$$STOP_IO (BAS$K_IOERR_REC);
; 1301          2009  3          END;
; 1302          2010  3
; 1303          2011  3  !+
; 1304          2012  3  ! Update pointer and the number of bytes left to move to the array.
; 1305          2013  3  !-
; 1306          2014  3
; 1307          2015  3      SOURCE = .SOURCE + .LEN;
; 1308          2016  3      REMAINING_BYTES = .REMAINING_BYTES - .LEN;
; 1309          2017  3
; 1310          2018  2      END;                                        ! end of loop through values
; 1311          2019  2
; 1312          2020  2  !+
; 1313          2021  2  ! Done with this I/O channel.
; 1314          2022  2  !-
; 1315          2023  2      BAS$$CB_POP ();
; 1316          2024  1      END;                                        ! end of BAS$$WHOLE_VA_STORE


                                 0FFC 00000          .ENTRY  BAS$$WHOLE_VA_STORE, Save R2,R3,R4,R5,R6,-  ; 1710
                                                             R7,R8,R9,R10,R11                            ;
              5E         A4   AE   9E 00002          MOVAB   -92(SP), SP                                 ;
                         58   AE   D4 00006          CLRL    SAVE_CCB                                    ; 1746
              6D       02AC   CF   DE 00009          MOVAL   32$, (FP)                                   ;
                         08   AC   DD 0000E          PUSHL   VALUE                                       ; 1776
              57         04   AC   D0 00011          MOVL    DESCRIP, R7                                 ; 1783
     50         0C   A7  F8   A7   C1 00015          ADDL3   -8(R7), 12(R7), R0                          ;
       00000200  8F               50   D1 0001B          CMPL    R0, #512                                    ;
                              06   18 00022          BGEQ    1$                                          ;
              56         0C   A7   D0 00024          MOVL    12(R7), REMAINING_BYTES                     ; 1785
                              09   11 00028          BRB     2$                                          ;
     56 00000200  8F         F8   A7   C3 0002A 1$:      SUBL3   -8(R7), #512, REMAINING_BYTES               ; 1787
                 52         FC   A7   D0 00033 2$:      MOVL    -4(R7), CHAN                                ; 1792
                              50   D4 00037          CLRL    R0                                          ; 1798
                  00000000G   00   16 00039          JSB     BAS$$CB_PUSH                                ;
            5C   AE               5B   D0 0003F          MOVL    CCB, SAVE_CCB                               ; 1799
                 0B         FC   AB   E8 00043          BLBS    -4(CCB), 3$                                 ; 1801
                 7E        00G   8F   9A 00047          MOVZBL  #BAS$K_VIRARROPE, -(SP)                     ;
       00000000G  00               01   FB 0004B          CALLS   #1, BAS$$STOP                               ;
                 05         C4   AB   91 00052 3$:      CMPB    -60(CCB), #5                                ; 1809
                              0B   13 00056          BEQL    4$                                          ;
                 7E       00G  8F  9A 00058        MOVZBL  #BAS$K_VIRARRDIS, -(SP)                  :
       00000000G 00             01  FB 0005C        CALLS   #1, BAS$$STOP_IO                         :
    0B        FF AB             01  E1 00063 4$:    BBC     #1, -1(CCB), 5$                          : 1815
                 7E       00G  8F  9A 00068        MOVZBL  #BAS$K_ILLOPE, -(SP)                     :
       00000000G 00             01  FB 0006C        CALLS   #1, BAS$$STOP_IO                         :
            0200 8F        20  AB  B1 00073 5$:    CMPW    32(CCB), #512                            : 1821
                                0B  13 00079        BEQL    6$                                       :
                 7E       00G  8F  9A 0007B        MOVZBL  #BAS$K_VIRBUFTOO, -(SP)                  :
       00000000G 00             01  FB 0007F        CALLS   #1, BAS$$STOP_IO                         :
    0B        FC AB             02  E1 00086 6$:    BBC     #2, -4(CCB), 7$                          : 1827
                 7E       00G  8F  9A 0008B        MOVZBL  #BAS$K_ILLILLACC, -(SP)                  :
       00000000G 00             01  FB 0008F        CALLS   #1, BAS$$STOP_IO                         :
              FF AB             01  88 00096 7$:    BISB2   #1, -1(CCB)                              : 1832
              1E AB             01  90 0009A        MOVB    #1, 30(CCB)                              : 1836
              06 AB        80  8F  88 0009E        BISB2   #128, 6(CCB)                             : 1841
                           A4  AB  D5 000A3        TSTL    -92(CCB)                                 : 1847
                                06  12 000A6        BNEQ    8$                                       :
              A4 AB     0000V  CF  9E 000A8        MOVAB   BAS$$VA_CLOSE, -92(CCB)                  :
                 50     0000V  CF  9E 000AE 8$:    MOVAB   BAS$$VA_CLOSE, R0                        : 1849
                 50        A4  AB  D1 000B3        CMPL    -92(CCB), R0                             :
                                0B  13 000B7        BEQL    9$                                       :
                 7E       00G  8F  9A 000B9        MOVZBL  #BAS$K_PROLOSSOR, -(SP)                  :
       00000000G 00             01  FB 000BD        CALLS   #1, BAS$$STOP_IO                         :
                           E0  AB  D5 000C4 9$:    TSTL    -32(CCB)                                 : 1856
                                65  13 000C7        BEQL    14$                                      :
    60        FE AB             03  E1 000C9        BBC     #3, -2(CCB), 14$                         : 1859
                                5B  DD 000CE        PUSHL   CCB                                      : 1862
       00000000G 00             01  FB 000D0        CALLS   #1, SYS$PUT                              :
                 58             50  D0 000D7        MOVL    R0, PUT_STATUS                           :
       00010651 8F              58  D1 000DA        CMPL    PUT_STATUS, #67153                       : 1864
                                07  12 000E1        BNEQ    10$                                      :
       00000000G 00             00  FB 000E3        CALLS   #0, BAS$$SIGNAL_CTRLC                    : 1866
                 3D             58  E8 000EA 10$:   BLBS    PUT_STATUS, 13$                          : 1868
       000182DA 8F              58  D1 000ED 11$:   CMPL    PUT_STATUS, #99034                       : 1871
                                27  12 000F4        BNEQ    12$                                      :
                                5B  DD 000F6        PUSHL   CCB                                      : 1873
       00000000G 00             01  FB 000F8        CALLS   #1, SYS$WAIT                             :
                                5B  DD 000FF        PUSHL   CCB                                      : 1874
       00000000G 00             01  FB 00101        CALLS   #1, SYS$PUT                              :
                 58             50  D0 00108        MOVL    R0, PUT_STATUS                           :
       00010651 8F              58  D1 0010B        CMPL    PUT_STATUS, #67153                       : 1876
                                D9  12 00112        BNEQ    11$                                      :
       00000000G 00             00  FB 00114        CALLS   #0, BAS$$SIGNAL_CTRLC                    : 1878
                                D0  11 0011B        BRB     11$                                      : 1871
                 0A             58  E8 0011D 12$:   BLBS    PUT_STATUS, 13$                          : 1881
                 7E             01  CE 00120        MNEGL   #1, -(SP)                                :
       00000000G 00             01  FB 00123        CALLS   #1, BAS$$STOP_IO                         :
              FE AB             08  8A 0012A 13$:   BICB2   #8, -2(CCB)                              : 1883
    50        F8 A7        0C  A7  C1 0012E 14$:   ADDL3   12(R7), -8(R7), R0                       : 1892
              50 00000200      8F  C6 00134        DIVL2   #512, R0                                 :
              08 AE        01  A0  9E 0013B        MOVAB   1(R0), 8(SP)                             :
                                5A  D4 00140        CLRL    BLKCNT                                   : 1975
                          0166  31 00142        BRW     31$                                      :
              E0 AB             5A  D0 00145 15$:   MOVL    BLKCNT, -32(CCB)                         : 1894
                                5B  DD 00149        PUSHL   CCB                                      : 1895
       00000000G 00             01  FB 0014B        CALLS   #1, SYS$GET                              :
                 59             50  D0 00152        MOVL    R0, GET_STATUS                           :
```

```
                               00010651   8F           59 D1 00155        CMPL    GET_STATUS, #67153                        ; 1897
                                                       07 12 0015C        BNEQ    16$
                              00000000G   00           00 FB 0015E        CALLS   #0, BAS$$SIGNAL_CTRLC                     ; 1899
                               0001827A   8F           59 D1 00165 16$:   CMPL    GET_STATUS, #98938                        ; 1905
                                                       09 13 0016C        BEQL    17$
                               00018049   8F           59 D1 0016E        CMPL    GET_STATUS, #98377
                                                       66 12 00175        BNEQ    20$
                                     E4   AB     E0    AB D1 00177 17$:   CMPL    -32(CCB), -28(CCB)                        ; 1919
                                                       4B 15 0017C        BLEQ    19$
   0050   8F              00              6E           00 2C 0017E        MOVC5   #0, (SP), #0, #80, $RMS_PTR               ; 1922
                                                 0C    AE    00185
                                     0C   AE   5003    8F B0 00187        MOVW    #20483, $RMS_PTR
                                     22   AE           02 90 0018D        MOVB    #2, $RMS_PTR+22
                                     2B   AE           02 90 00191        MOVB    #2, $RMS_PTR+31
                  1C      AE         E0   AB     E4    AB C3 00195        SUBL3   -28(CCB), -32(CCB), FAB_BLOCK+16          ; 1923
                                     0E   AE     D0    AB B0 0019C        MOVW    -48(CCB), FAB_BLOCK+2                     ; 1924
                                     E8   AB     0C    AE 9E 001A1        MOVAB   FAB_BLOCK, -24(CCB)                       ; 1925
                                     08   AB           01 D0 001A6        MOVL    #1, 8(CCB)                                ; 1926
                                                 0C    AE 9F 001AA        PUSHAB  FAB_BLOCK                                 ; 1927
                              00000000G   00           01 FB 001AD        CALLS   #1, SYS$EXTEND
                                          0A           50 E8 001B4        BLBS    EXTEND_STATUS, 18$                        ; 1929
                                          7E           01 CE 001B7        MNEGL   #1, -(SP)
                              00000000G   00           01 FB 001BA        CALLS   #1, BAS$$STOP_IO
                                     E4   AB     1C    AE C0 001C1 18$:   ADDL2   FAB_BLOCK+16, -28(CCB)                    ; 1931
                                                 E8    AB D4 001C6        CLRL    -24(CCB)                                  ; 1932
     20   AB              00              6E           00 2C 001C9 19$:   MOVC5   #0, (SP), #0, 32(CCB), @36(CCB)           ; 1939
                                                 24    BB    001CF
                                     22   AB     20    AB B0 001D1        MOVW    32(CCB), 34(CCB)                          ; 1940
                                     28   AB     24    AB D0 001D6        MOVL    36(CCB), 40(CCB)                          ; 1941
                                                       40 11 001DB        BRB     23$                                       ; 1905
                                          3D           59 E8 001DD 20$:   BLBS    GET_STATUS, 23$                           ; 1946
                               000182DA   8F           59 D1 001E0 21$:   CMPL    GET_STATUS, #99034                        ; 1953
                                                       27 12 001E7        BNEQ    22$
                                                       5B DD 001E9        PUSHL   CCB                                       ; 1955
                              00000000G   00           01 FB 001EB        CALLS   #1, SYS$WAIT
                                                       5B DD 001F2        PUSHL   CCB                                       ; 1956
                              00000000G   00           01 FB 001F4        CALLS   #1, SYS$GET
                                          59           50 D0 001FB        MOVL    R0, GET_STATUS
                               00010651   8F           59 D1 001FE        CMPL    GET_STATUS, #67153                        ; 1958
                                                       D9 12 00205        BNEQ    21$
                              00000000G   00           00 FB 00207        CALLS   #0, BAS$$SIGNAL_CTRLC                     ; 1960
                                                       D0 11 0020E        BRB     21$                                       ; 1953
                                          0A           59 E8 00210 22$:   BLBS    GET_STATUS, 23$                           ; 1964
                                          7E           01 CE 00213        MNEGL   #1, -(SP)
                              00000000G   00           01 FB 00216        CALLS   #1, BAS$$STOP_IO
                                          50           56 D0 0021D 23$:   MOVL    REMAINING_BYTES, R0                       ; 1974
                               00000200   8F           50 D1 00220        CMPL    R0, #512
                                                       05 15 00227        BLEQ    24$
                                          50   0200    8F 3C 00229        MOVZWL  #512, R0
                                     04   AE           50 D0 0022E 24$:   MOVL    R0, LEN
                                          01           5A D1 00232        CMPL    BLKCNT, #1                                ; 1976
                                                       06 12 00235        BNEQ    25$
                                          50     F8    A7 D0 00237        MOVL    -8(R7), R0
                                                       02 11 0023B        BRB     26$
                                                       50 D4 0023D 25$:   CLRL    R0
                  28    BB40         00   BE     04    AE 28 0023F 26$:   MOVC3   LEN, @SOURCE, @40(CCB)[R0]                ; 1975
                                                       5B DD 00247        PUSHL   CCB                                       ; 1982
                  00000000G  00            01  FB 00249           CALLS   #1, SYS$PUT                          ;
                             58            50  D0 00250           MOVL    R0, PUT_STATUS                       ;
                  00010651   8F            58  D1 00253           CMPL    PUT_STATUS, #67153                   ; 1984
                                           07  12 0025A           BNEQ    27$                                  ;
                  00000000G  00            00  FB 0025C           CALLS   #0, BAS$$SIGNAL_CTRLC                ; 1986
                             3D            58  E8 00263 27$:      BLBS    PUT_STATUS, 30$                      ; 1994
                  000182DA   8F            58  D1 00266 28$:      CMPL    PUT_STATUS, #99034                   ; 1997
                                           27  12 0026D           BNEQ    29$                                  ;
                                           5B  DD 0026F           PUSHL   CCB                                  ; 1999
                  00000000G  00            01  FB 00271           CALLS   #1, SYS$WAIT                         ;
                                           5B  DD 00278           PUSHL   CCB                                  ; 2000
                  00000000G  00            01  FB 0027A           CALLS   #1, SYS$PUT                          ;
                             58            50  D0 00281           MOVL    R0, PUT_STATUS                       ;
                  00010651   8F            58  D1 00284           CMPL    PUT_STATUS, #67153                   ; 2002
                                           D9  12 0028B           BNEQ    28$                                  ;
                  00000000G  00            00  FB 0028D           CALLS   #0, BAS$$SIGNAL_CTRLC                ; 2004
                                           D0  11 00294           BRB     28$                                  ; 1997
                             0A            58  E8 00296 29$:      BLBS    PUT_STATUS, 30$                      ; 2008
                             7E            01  CE 00299           MNEGL   #1, -(SP)                            ;
                  00000000G  00            01  FB 0029C           CALLS   #1, BAS$$STOP_IO                     ;
                             6E       04   AE  C0 002A3 30$:      ADDL2   LEN, SOURCE                          ; 2015
                             56       04   AE  C2 002A7           SUBL2   LEN, REMAINING_BYTES                 ; 2016
      FE93           5A      01       08   AE  F1 002AB 31$:      ACBL    8(SP), #1, BLKCNT, 15$               ; 1892
                                00000000G  00  16 002B2           JSB     BAS$$CB_POP                          ; 2023
                                               04 002B8           RET                                          ; 2024
                                             0000 002B9 32$:      .WORD   Save nothing                         ; 1746
                             50       08   AC  D0 002BB           MOVL    8(AP), R0                            ;
                             50       04   A0  D0 002BF           MOVL    4(R0), R0                            ;
                                      FC   A0  9F 002C3           PUSHAB  SAVE_CCB                             ;
                                           01  DD 002C6           PUSHL   #1                                   ;
                                           5E  DD 002C8           PUSHL   SP                                   ;
                             7E       04   AC  7D 002CA           MOVQ    4(AP), -(SP)                         ;
                      0000V  CF            03  FB 002CE           CALLS   #3, HANDLER                          ;
                                               04 002D3           RET                                          ;

; Routine Size:  724 bytes,    Routine Base:  _BAS$CODE + 0647


; 1317         2025  1
```

```
1 ROUTINE BAS$$VA_CLOSE                                      ! Close a virtual array
1     : CALL_CCB NOVALUE =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Handle the closing of a virtual array.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !
1 !       Writes out the last I/O buffer (if it has been modified).
1 !
1 !--
1
2    BEGIN
2
2    EXTERNAL REGISTER
2        CCB : REF BLOCK [, BYTE];
2
2    LOCAL
2        PUT_STATUS;                                         ! Status of last RMS PUT
2
2 !+
2 ! Record access will always be by key.
2 !-
2    CCB [RAB$B_RAC] = RAB$C_KEY;
2
2 !+
2 ! If the buffer is dirty, write it out.
2 !-
2
3    IF (.CCB [LUB$V_OUTBUF_DR])
2    THEN
3        BEGIN
3        PUT_STATUS = $PUT (RAB = .CCB);
3
3        IF .PUT_STATUS EQL RMS$_CONTROLC
3        THEN
3            BAS$$SIGNAL_CTRLC ();
```

```
; 1376          2083  4           IF ( NOT .PUT_STATUS)
; 1377          2084  3           THEN
; 1378          2085  4               BEGIN
; 1379          2086  4   !+
; 1380          2087  4   ! Worry about RMS RSA error.
; 1381          2088  4   !-
; 1382          2089  4
; 1383          2090  4               WHILE (.PUT_STATUS EQL RMS$_RSA) DO
; 1384          2091  5                   BEGIN
; 1385          2092  5                   $WAIT (RAB = .CCB);
; 1386          2093  5                   PUT_STATUS = $PUT (RAB = .CCB);
; 1387          2094  5
; 1388          2095  5                   IF .PUT_STATUS EQL RMS$_CONTROLC
; 1389          2096  5                   THEN
; 1390          2097  5                       BAS$$SIGNAL_CTRLC ();
; 1391          2098  5
; 1392          2099  4                   END;
; 1393          2100  4
; 1394          2101  4               IF ( NOT .PUT_STATUS) THEN BAS$$STOP_IO (BAS$K_IOERR_REC);
; 1395          2102  4
; 1396          2103  3               END;
; 1397          2104  3
; 1398          2105  3           CCB [LUB$V_OUTBUF_DR] = 0;
; 1399          2106  2           END;
; 1400          2107  2
; 1401          2108  1       END;                                        ! end of BAS$$VA_CLOSE



                                            001C 00000 BAS$$VA_CLOSE:
                                                              .WORD     Save R2,R3,R4                              ; 2026
                          54 00000000G  00    9E 00002        MOVAB     SYS$PUT, R4                                ;
                          53 00000000G  00    9E 00009        MOVAB     BAS$$SIGNAL_CTRLC, R3                      ;
                   1E     AB            01    90 00010        MOVB      #1, 30(CCB)                                ; 2068
          50       FE     AB            03    E1 00014        BBC       #3, -2(CCB), 5$                            ; 2074
                                        5B    DD 00019        PUSHL     CCB                                        ; 2077
                          64            01    FB 0001B        CALLS     #1, SYS$PUT                                ;
                          52            50    D0 0001E        MOVL      R0, PUT_STATUS                             ;
             00010651     8F            52    D1 00021        CMPL      PUT_STATUS, #67153                         ; 2079
                                        03    12 00028        BNEQ      1$                                         ;
                          63            00    FB 0002A        CALLS     #0, BAS$$SIGNAL_CTRLC                      ; 2081
                          35            52    E8 0002D 1$:    BLBS      PUT_STATUS, 4$                             ; 2083
             000182DA     8F            52    D1 00030 2$:    CMPL      PUT_STATUS, #99034                         ; 2090
                                        1F    12 00037        BNEQ      3$                                         ;
                                        5B    DD 00039        PUSHL     CCB                                        ; 2092
             00000000G    00            01    FB 0003B        CALLS     #1, SYS$WAIT                               ;
                                        5B    DD 00042        PUSHL     CCB                                        ; 2093
                          64            01    FB 00044        CALLS     #1, SYS$PUT                                ;
                          52            50    D0 00047        MOVL      R0, PUT_STATUS                             ;
             00010651     8F            52    D1 0004A        CMPL      PUT_STATUS, #67153                         ; 2095
                                        DD    12 00051        BNEQ      2$                                         ;
                          63            00    FB 00053        CALLS     #0, BAS$$SIGNAL_CTRLC                      ; 2097
                                        D8    11 00056        BRB       2$                                         ; 2090
                          0A            52    E8 00058 3$:    BLBS      PUT_STATUS, 4$                             ; 2101
                          7E            01    CE 0005B        MNEGL     #1, -(SP)                                  ;
```

```
                    00000000G  00          01  FB 0005E        CALLS   #1, BAS$$STOP_IO                              ;
                           FE  AB          08  8A 00065 4$:    BICB2   #8, -2(CCB)                                   ; 2105
                                               04 00069 5$:    RET                                                   ; 2108

; Routine Size:  106 bytes,    Routine Base:  _BAS$CODE + 091B
```

```
1 ROUTINE HANDLER (                                       ! POP CCB on UNWIND
1         SIG,                                            ! signal args
1         MECH,                                           ! mechanism args
1         ENBL                                            ! variables
1     ) =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       POP the CCB if one of the routines which does a PUSH gets an
1 !       error.  This is handled here rather than in the BASIC handler
1 !       so that the address of the frame of the caller does not have
1 !       to be passed all the way down to this module.  Also, we wish
1 !       to mark that there is no buffer in memory.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       SIG.rl.a        Address of the signal vector.  This contains
1 !                       the condition.
1 !       MECH.rl.a       Address of the mechanism vector.  This contains
1 !                       the status of the frame that signalled.
1 !       ENBL.rl.a       Address of the enable vector.  This contains
1 !                       a pointer to the CCB, or 0.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 !
1 !       NONE
1 !
1 ! COMPLETION CODES:
1 !
1 !       Always SS$_RESIGNAL, but this is ingored when we are
1 !       unwinding.
1 !
1 ! SIDE EFFECTS:
1 !
1 !       Usually calls BAS$$CB_POP to complete the I/O.
1 !       Also marks the buffer empty.
1 !
1 !--
1
2     BEGIN
2
2     MAP
2         SIG : REF VECTOR,                               ! signal vector
2         MECH : REF VECTOR,                              ! mechanism vector
2         ENBL : REF VECTOR;                              ! enable vector
2
2     GLOBAL REGISTER
2         CCB = K_CCB_REG : REF BLOCK [, BYTE];
```

```
; 1460          2166  2
; 1461          2167  2        BIND
; 1462          2168  2            SAVE_CCB = .ENBL [1];
; 1463          2169  2
; 1464          2170  2    !+
; 1465          2171  2    ! If this is the unwind condition, restore the CCB.
; 1466          2172  2    ! Mark that there is no buffer in memory.
; 1467          2173  2    !-
; 1468          2174  2
; 1469          2175  3        IF ((LIB$MATCH_COND (SIG [1], %REF (SS$_UNWIND))) AND (.SAVE_CCB NEQA 0))
; 1470          2176  2        THEN
; 1471          2177  3            BEGIN
; 1472          2178  3            BAS$$CB_GET ();
; 1473          2179  3
; 1474          2180  4            IF (.CCB EQLA .SAVE_CCB)
; 1475          2181  3            THEN
; 1476          2182  4                BEGIN
; 1477          2183  4                CCB [LUB$L_LOG_RECNO] = 0;
; 1478          2184  4                CCB [LUB$V_OUTBUF_DR] = 0;
; 1479          2185  4                BAS$$CB_POP ();
; 1480          2186  4                RETURN (SS$_RESIGNAL);
; 1481          2187  4                END
; 1482          2188  3            ELSE
; 1483          2189  3
; 1484          2190  3                IF (.CCB NEQA 0) THEN LIB$STOP (OTS$_FATINTERR);
; 1485          2191  3
; 1486          2192  2            END;
; 1487          2193  2
; 1488          2194  2    !+
; 1489          2195  2    ! We do not recognize the signal, pass it without comment.
; 1490          2196  2    !-
; 1491          2197  2        RETURN (SS$_RESIGNAL);
; 1492          2198  1        END;                                        ! of HANDLER



                                        0804 00000 HANDLER:.WORD   Save R2,R11                                   ; 2109
                     52         0C   AC   D0 00002         MOVL    ENBL, R2                                      ; 2168
                     7E       0920   8F   3C 00006         MOVZWL  #2336, -(SP)                                  ; 2175
                                     5E   DD 0000B         PUSHL   SP                                            ;
            7E       04     AC       04   C1 0000D         ADDL3   #4, SIG, -(SP)                                ;
              00000000G     00       02   FB 00012         CALLS   #2, LIB$MATCH_COND                            ;
                            31       50   E9 00019         BLBC    R0, 2$                                        ;
                                04   B2   D5 0001C         TSTL    @4(R2)                                        ;
                                     2C   13 0001F         BEQL    2$                                            ;
                        00000000G    00   16 00021         JSB     BAS$$CB_GET                                   ; 2178
                     04     B2       5B   D1 00027         CMPL    CCB, @4(R2)                                   ; 2180
                                     0F   12 0002B         BNEQ    1$                                            ;
                                E0   AB   D4 0002D         CLRL    -32(CCB)                                      ; 2183
                     FE     AB       08   8A 00030         BICB2   #8, -2(CCB)                                   ; 2184
                        00000000G    00   16 00034         JSB     BAS$$CB_POP                                   ; 2185
                                     11   11 0003A         BRB     2$                                            ; 2186
                                     5B   D5 0003C 1$:     TSTL    CCB                                           ; 2190
                                     0D   13 0003E         BEQL    2$                                            ;
                        00000000G    8F   DD 00040         PUSHL   #OTS$_FATINTERR                               ;
```

```
                                              N 14
BAS$$VIRT_IO                                 16-Sep-1984 01:28:00    VAX-11 Bliss-32 V4.0-742              Page  43
1-027                                        14-Sep-1984 11:56:46    [BASRTL.SRC]BASVIRTIO.B32;1              (8)

                          00000000G  00            01  FB 00046          CALLS   #1, LIB$STOP                        ;
                                     50      0918  8F  3C 0004D 2$:      MOVZWL  #2328, R0                           ; 2197
                                                       04 00052          RET                                         ; 2198

; Routine Size:  83 bytes,    Routine Base:  _BAS$CODE + 0985


; 1493          2199  1 END                                            ! end of module BAS$VIRT_IO
; 1494          2200  1
; 1495          2201  0 ELUDOM




;                                PSECT SUMMARY
;
;       Name                         Bytes                   Attributes
;
;   _BAS$CODE                            2520  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]STARLET.L32;1         9776         70         0            581          00:01.1



;                                 COMMAND QUALIFIERS

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

; Size:          2520 code + 0 data bytes
; Run Time:         00:44.1
; Elapsed Time:     01:31.2
; Lines/CPU Min:     2991
; Lexemes/CPU-Min: 26594
; Memory Used:  251 pages
; Compilation Complete
```

0033 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

BASVIRTUA
LIS

BASUDFWL
LIS

BASUNLOCK
LIS

BASVECTOR
LIS

BASVAL
LIS

BASVIRTIO
LIS

BASUNWIND
LIS

BASUPDATE
LIS

BASVECTR2
LIS