```
BBBBBBBBBBBB        AAAAAAAAA       SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBBBBBBBBBBB        AAAAAAAAA       SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBBBBBBBBBBB        AAAAAAAAA       SSSSSSSSSSS  RRRRRRRRRRR     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    RRRRRRRRRRR       TTT        LLL
BBBBBBBBBBBB       AAA         AAA     SSSSSSSSS    RRRRRRRRRRR       TTT        LLL
BBBBBBBBBBBB       AAA         AAA     SSSSSSSSS    RRRRRRRRRRR       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  SSSSSSSSSSS  RRR         RRR    TTT        LLLLLLLLLLLLLLL
BBBBBBBBBBBB       AAA         AAA  SSSSSSSSSSS  RRR         RRR    TTT        LLLLLLLLLLLLLLL
BBBBBBBBBBBB       AAA         AAA  SSSSSSSSSSS  RRR         RRR    TTT        LLLLLLLLLLLLLLL
```

FILEID**BASEXITHA

```
BBBBBBBB      AAAAAA      SSSSSSSS  EEEEEEEEEE  XX      XX    IIIIII    TTTTTTTTTT  HH      HH    AAAAAA
BBBBBBBB      AAAAAA      SSSSSSSS  EEEEEEEEEE  XX      XX    IIIIII    TTTTTTTTTT  HH      HH    AAAAAA
BB      BB  AA      AA  SS          EE          XX      XX      II        TT      HH      HH  AA      AA
BB      BB  AA      AA  SS          EE          XX      XX      II        TT      HH      HH  AA      AA
BB      BB  AA      AA  SS          EE            XX  XX        II        TT      HH      HH  AA      AA
BB      BB  AA      AA  SS          EE            XX  XX        II        TT      HH      HH  AA      AA
BBBBBBBB    AA      AA    SSSSSS    EEEEEEEE        XX          II        TT      HHHHHHHHHH  AA      AA
BBBBBBBB    AA      AA    SSSSSS    EEEEEEEE        XX          II        TT      HHHHHHHHHH  AA      AA
BB      BB  AAAAAAAAAA          SS  EE            XX  XX        II        TT      HH      HH  AAAAAAAAAA
BB      BB  AAAAAAAAAA          SS  EE            XX  XX        II        TT      HH      HH  AAAAAAAAAA
BB      BB  AA      AA          SS  EE          XX      XX      II        TT      HH      HH  AA      AA    ....
BB      BB  AA      AA          SS  EE          XX      XX      II        TT      HH      HH  AA      AA    ....
BBBBBBBB    AA      AA  SSSSSSSS    EEEEEEEEEE  XX      XX    IIIIII      TT      HH      HH  AA      AA    ....
BBBBBBBB    AA      AA  SSSSSSSS    EEEEEEEEEE  XX      XX    IIIIII      TT      HH      HH  AA      AA    ....
```

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

```
;     1          0001  0 MODULE BAS$$EXIT_HANDL (                      ! BASIC exit handler
;     2          0002  0                       IDENT = '1-016'         ! File: BASEXITHA.B32 Edit: PL1016
;     3          0003  0                       ) =
;     4          0004  1 BEGIN
;     5          0005  1 !
;     6          0006  1 !**************************************************************************
;     7          0007  1 !*                                                                        *
;     8          0008  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                               *
;     9          0009  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                 *
;    10          0010  1 !*  ALL RIGHTS RESERVED.                                                   *
;    11          0011  1 !*                                                                        *
;    12          0012  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
;    13          0013  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
;    14          0014  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
;    15          0015  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
;    16          0016  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
;    17          0017  1 !*  TRANSFERRED.                                                           *
;    18          0018  1 !*                                                                        *
;    19          0019  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
;    20          0020  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
;    21          0021  1 !*  CORPORATION.                                                           *
;    22          0022  1 !*                                                                        *
;    23          0023  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
;    24          0024  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
;    25          0025  1 !*                                                                        *
;    26          0026  1 !*                                                                        *
;    27          0027  1 !**************************************************************************
;    28          0028  1
;    29          0029  1 !++
;    30          0030  1 ! FACILITY: BASIC support library - Exit handler
;    31          0031  1 !
;    32          0032  1 ! ABSTRACT:
;    33          0033  1 !
;    34          0034  1 !       This module is used when the image exits to do
;    35          0035  1 !       BASIC post processing.  It purges I/O buffers
;    36          0036  1 !       and closes files with proper disposition.
;    37          0037  1 !
;    38          0038  1 ! ENVIRONMENT: User access mode; mixture of AST level or not.
;    39          0039  1 !
;    40          0040  1 ! Author:       John Sauter, Creation date: 23-JAN-1979
;    41          0041  1 !
;    42          0042  1 ! MODIFIED BY:
;    43          0043  1 !
;    44          0044  1 ! 1-001 - Original from FOROPEN.  JBS 23-JAN-1979
;    45          0045  1 ! 1-002 - Call OTS$$PURGE_IOBU to flush any "dirty" buffer.  JBS 24-JAN-1979
;    46          0046  1 ! 1-003 - Move call to OTS$$PURGE_IOBU to OTS$CLOSE_FILE.  JBS 24-JAN-1979
;    47          0047  1 ! 1-004 - Change linkage for OTS$PUSH_CCB to JSB_CB_PUSH and for
;    48          0048  1 !           OTS$POP_CCB to JSB_CB_POP.  JBS 25-JAN-1979
;    49          0049  1 ! 1-005 - Use two dollar signs for non-user entries.  JBS 26-JAN-1979
;    50          0050  1 ! 1-006 - Add OTS$$CLOSE_ALL.  JBS 04-JUN-1979
;    51          0051  1 ! 1-007 - Change to BASIC-specific exit handler.  JBS 16-AUG-1979
;    52          0052  1 ! 1-008 - Call BAS$$PUR_IO_CLO to flush all buffers.  JBS 20-AUG-1979
;    53          0053  1 ! 1-009 - Make BAS$$CLOSE_ALL global, for BAS$$RUN_INIT.  JBS 21-AUG-1979
;    54          0054  1 ! 1-010 - Signal CLOSE errors, but make the severity "warning" so we
;    55          0055  1 !           don't lose control.  JBS 24-AUG-1979
;    56          0056  1 ! 1-011 - Do explicit signalling of CLOSE errors, since OTS$$CLOSE_FILE
;    57          0057  1 !           doesn't.  JBS 27-AUG-1979
```

```
;    58          0058  1 ! 1-012 - Give CLOSE_ALL an optional parameter, so we can close all of the
;    59          0059  1 !           streams connected to a base file.  JBS 28-SEP-1979
;    60          0060  1 ! 1-013 - Clear BAS$$L_XIT_LOCK upon entry to the exit handler.  This
;    61          0061  1 !         allows user exit handlers to perform I/O, and get the proper
;    62          0062  1 !         cleanup upon leaving.
;    63          0063  1 ! 1-014 - If There is a file X that Y and Z have connected to ( via open
;    64          0064  1 !         clause CONNECT) then close Y and Z first and then close X.
;    65          0065  1 !         FM 12-aug-81.
;    66          0066  1 ! 1-015 - LIB$STOP should be declared EXTERNAL.  PLL 20-NOV-81
;    67          0067  1 ! 1-016 - Edit 1-014 breaks virtual files.  BAS$$CLOSE_ALL no longer
;    68          0068  1 !         tried to close them if they were open because LUB$V_M_STR_C
;    69          0069  1 !         was not set.  PLL 24-Feb-82
;    70          0070  1 !--
;    71          0071  1
;    72          0072  1 !<BLF/PAGE>
```

```
;    74          0073  1 !
;    75          0074  1 ! SWITCHES:
;    76          0075  1 !
;    77          0076  1
;    78          0077  1 SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
;    79          0078  1
;    80          0079  1 !
;    81          0080  1 ! LINKAGES:
;    82          0081  1 !
;    83          0082  1
;    84          0083  1 REQUIRE 'RTLIN:OTSLNK';                         ! Define all linkages
;    85          0512  1
;    86          0513  1 !
;    87          0514  1 ! TABLE OF CONTENTS:
;    88          0515  1 !
;    89          0516  1
;    90          0517  1 FORWARD ROUTINE
;    91          0518  1     BAS$$DECL_EXITH : NOVALUE,                  ! Declare EXIT handler
;    92          0519  1     EXIT_HANDLER : NOVALUE,                     ! Exit Handler
;    93          0520  1     BAS$$CLOSE_ALL : NOVALUE,                   ! Close all files
;    94          0521  1     TRY_TO_CLOSE : CALL_CCB NOVALUE,            ! Subroutine for EXIT_HANDLER
;    95          0522  1     CLOSE_HANDLER;                              ! Handler for CLOSE errors
;    96          0523  1
;    97          0524  1 !
;    98          0525  1 ! INCLUDE FILES:
;    99          0526  1 !
;   100          0527  1
;   101          0528  1 REQUIRE 'RTLML:OTSLUB';                         ! logical Unit Block definitions
;   102          0668  1
;   103          0669  1 REQUIRE 'RTLIN:OTSMAC';                         ! macros
;   104          0863  1
;   105          0864  1 REQUIRE 'RTLIN:BASIOERR';                       ! I/O error codes
;   106          0917  1
;   107          0918  1 REQUIRE 'RTLIN:RTLPSECT';                       ! Define DELCARE_PSECTS macro
;   108          1013  1
;   109          1014  1 LIBRARY 'RTLSTARLE';                           ! STARLET library for macros and symbols
;   110          1015  1
;   111          1016  1 !
;   112          1017  1 ! MACROS:
;   113          1018  1 !
;   114          1019  1 !       NONE
;   115          1020  1 !
;   116          1021  1 ! EQUATED SYMBOLS:
;   117          1022  1 !
;   118          1023  1 !       NONE
;   119          1024  1 !
;   120          1025  1 ! PSECT DECLARATIONS:
;   121          1026  1 !
;   122          1027  1 DECLARE_PSECTS (BAS);                           ! declare PSECTs for BAS$ facility
;   123          1028  1 !
;   124          1029  1 ! OWN STORAGE:
;   125          1030  1 !
;   126          1031  1
;   127          1032  1 OWN
;   128          1033  1     EXIT_REASON,                                ! VMS stuffs with reason for exiting
;   129          1034  1     EXIT_BLOCK : VECTOR [4] INITIAL (0,
;   130          1035  1                                                 ! Filled in by VMS with forward link to next EXIT control block
```

```
;  131          1036  1          0,                              ! Set to EXIT_HANDLER if RTL sets up EXIT handler
;  132          1037  1          0,                              ! Set to arg count (1) if RTL sets up EXIT handler
;  133          1038  1          0);                             ! Set to EXIT_REASON if RTL sets up EXIT handler
;  134          1039  1
;  135          1040  1  GLOBAL
;  136          1041  1      BAS$$L_XIT_LOCK : INITIAL (0);      ! Clear if no handler linked yet
;  137          1042  1
;  138          1043  1  !
;  139          1044  1  ! (Used to make sure only one handler even if ASTs)
;  140          1045  1  !
;  141          1046  1  ! EXTERNAL REFERENCES:
;  142          1047  1  !
;  143          1048  1
;  144          1049  1  EXTERNAL LITERAL
;  145          1050  1      OTS$_FATINTERR;                     ! OTS Fatal Internal Error
;  146          1051  1
;  147          1052  1  EXTERNAL ROUTINE
;  148          1053  1      LIB$STOP : NOVALUE,                 ! Signal a fatal error
;  149          1054  1      BAS$$CB_PUSH : JSB_CB_PUSH NOVALUE, ! Load register CCB
;  150          1055  1      BAS$$CB_POP : JSB_CB_POP NOVALUE,   ! Done with register CCB
;  151          1056  1      BAS$$NEXT_LUN : NOVALUE,            ! Get next LUN that might be open
;  152          1057  1      BAS$$PUR_IO_CLO : NOVALUE,          ! Purge all I/O buffers
;  153          1058  1      OTS$$CLOSE_FILE : CALL_CCB,         ! Internal file closer
;  154          1059  1      BAS$$SIGNAL_IO : CALL_CCB NOVALUE;  ! Signal a BASIC I/O error
;  155          1060  1
```

```
;  157          1061  1 GLOBAL ROUTINE BAS$$DECL_EXITH                    ! Declare VMS EXIT handler
;  158          1062  1     : NOVALUE =                                    !
;  159          1063  1
;  160          1064  1 !++
;  161          1065  1 ! FUNCTIONAL DESCRIPTION:
;  162          1066  1 !
;  163          1067  1 !       Declares VMS EXIT handler for BASIC.
;  164          1068  1 !
;  165          1069  1 ! CALLING SEQUENCE:
;  166          1070  1 !
;  167          1071  1 !       IF (NOT .BAS$$L_XIT_LOCK) THEN BAS$$DECL_EXITH ()
;  168          1072  1 !
;  169          1073  1 ! FORMAL PARAMETERS:
;  170          1074  1 !
;  171          1075  1 !       NONE
;  172          1076  1 !
;  173          1077  1 ! IMPLICIT INPUTS:
;  174          1078  1 !
;  175          1079  1 !       NONE
;  176          1080  1 !
;  177          1081  1 ! IMPLICIT OUTPUTS:
;  178          1082  1 !
;  179          1083  1 !       NONE
;  180          1084  1 !
;  181          1085  1 ! ROUTINE VALUE:
;  182          1086  1 ! COMPLETION CODES:
;  183          1087  1 !
;  184          1088  1 !       NONE
;  185          1089  1 !
;  186          1090  1 ! SIDE EFFECTS:
;  187          1091  1 !
;  188          1092  1 !       Declares VMS EXIT handler.
;  189          1093  1 !--
;  190          1094  1
;  191          1095  2     BEGIN
;  192          1096  2
;  193          1097  2     LOCAL
;  194          1098  2         AST_STATUS,
;  195          1099  2         DCLEXH_STATUS;
;  196          1100  2
;  197          1101  2 !+
;  198          1102  2 ! We must disable ASTs to be sure that one and only one exit handler
;  199          1103  2 ! is declared for BASIC.
;  200          1104  2 !-
;  201          1105  2     AST_STATUS = $SETAST (ENBFLG = 0);
;  202          1106  2
;  203          1107  3     IF ( NOT .BAS$$L_XIT_LOCK)
;  204          1108  2     THEN
;  205          1109  3         BEGIN
;  206          1110  3 !+
;  207          1111  3 ! Initialize EXIT handler control block (must do at run time to be PIC)
;  208          1112  3 !-
;  209          1113  3         EXIT_BLOCK [1] = EXIT_HANDLER;              ! Adr. of EXIT handler to be called on EXIT
;  210          1114  3         EXIT_BLOCK [2] = 1;                         ! arg count
;  211          1115  3         EXIT_BLOCK [3] = EXIT_REASON;               ! adr. to store reason for EXIT
;  212          1116  3         DCLEXH_STATUS = $DCLEXH (DESBLK = EXIT_BLOCK);
;  213          1117  3         BAS$$L_XIT_LOCK = 1;
```

```
;  214          1118  3          END
;  215          1119  2      ELSE
;  216          1120  2          DCLEXH_STATUS = 1;
;  217          1121  2
;  218          1122  2      IF (.AST_STATUS EQL SS$_WASSET) THEN $SETAST (ENBFLG = 1);
;  219          1123  2
;  220          1124  2      IF ( NOT .DCLEXH_STATUS) THEN LIB$STOP (OTS$_FATINTERR);
;  221          1125  2
;  222          1126  2      RETURN
;  223          1127  1      END;


                                                        .TITLE  BAS$$EXIT_HANDL
                                                        .IDENT  \1-016\

                                                        .PSECT  _BAS$DATA,NOEXE,  PIC,2

                                                00000 EXIT_REASON:
                                                        .BLKB   4
                00000000  00000000  00000000  00000000  00004 EXIT_BLOCK:
                                                        .LONG   0, 0, 0, 0                              ;
                                                00000000  00014 BAS$$L_XIT_LOCK::
                                                        .LONG   0                                       ;

                                                        .EXTRN  OTS$_FATINTERR, LIB$STOP
                                                        .EXTRN  BAS$$CB_PUSH, BAS$$CB_POP
                                                        .EXTRN  BAS$$NEXT_LUN, BAS$$POR_IO_CLO
                                                        .EXTRN  OTS$$CLOSE_FILE
                                                        .EXTRN  BAS$$SIGNAL_IO, SYS$SETAST
                                                        .EXTRN  SYS$DCLEXH

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

                                                   003C 00000          .ENTRY  BAS$$DECL_EXITH, Save R2,R3,R4,R5       ; 1061
                                55 00000000G    00   9E 00002          MOVAB   SYS$SETAST, R5                          ;
                                54 00000000'    EF   9E 00009          MOVAB   BAS$$L_XIT_LOCK, R4                     ;
                                                7E   D4 00010          CLRL    -(SP)                                   ; 1105
                                65              01   FB 00012          CALLS   #1, SYS$SETAST                          ;
                                53              50   D0 00015          MOVL    R0, AST_STATUS                          ;
                                21              64   E8 00018          BLBS    BAS$$L_XIT_LOCK, 1$                     ; 1107
                          F4    A4      0000V   CF   9E 0001B          MOVAB   EXIT_HANDLER, EXIT_BLOCK+4              ; 1113
                          F8    A4              01   D0 00021          MOVL    #1, EXIT_BLOCK+8                        ; 1114
                          FC    A4        EC    A4   9E 00025          MOVAB   EXIT_REASON, EXIT_BLOCK+12              ; 1115
                                          F0    A4   9F 0002A          PUSHAB  EXIT_BLOCK                              ; 1116
                00000000G 00              01   FB 0002D          CALLS   #1, SYS$DCLEXH                          ;
                                52              50   D0 00034          MOVL    R0, DCLEXH_STATUS                       ;
                                64              01   D0 00037          MOVL    #1, BAS$$L_XIT_LOCK                     ; 1117
                                                03   11 0003A          BRB     2$                                      ; 1107
                                52              01   D0 0003C 1$:      MOVL    #1, DCLEXH_STATUS                       ; 1120
                                09              53   D1 0003F 2$:      CMPL    AST_STATUS, #9                          ; 1122
                                                05   12 00042          BNEQ    3$                                      ;
                                                01   DD 00044          PUSHL   #1                                      ;
                                65              01   FB 00046          CALLS   #1, SYS$SETAST                          ;
                                0D              52   E8 00049 3$:      BLBS    DCLEXH_STATUS, 4$                       ; 1124
                                      00000000G 8F   DD 0004C          PUSHL   #OTS$_FATINTERR                         ;
                00000000G 00                    01   FB 00052          CALLS   #1, LIB$STOP                            ;
                                                04   00059 4$:      RET                                             ; 1127
```

```
; Routine Size:  90 bytes,    Routine Base:  _BAS$CODE + 0000

;  224          1128  1
```

```
;  226          1129  1 ROUTINE EXIT_HANDLER (                          ! Exit Handler
;  227          1130  1         EXIT_REASON                             ! Reason
;  228          1131  1     ) : NOVALUE =
;  229          1132  1
;  230          1133  1 !++
;  231          1134  1 ! FUNCTIONAL DESCRIPTION:
;  232          1135  1 !
;  233          1136  1 !       This is the exit handler for BASIC.  Its only function is to
;  234          1137  1 !       purge I/O buffers and close all files.
;  235          1138  1 !
;  236          1139  1 !       Upon entry, it zeroes BAS$$L_XIT_LOCK so that user I/O in
;  237          1140  1 !       exit handlers can get properly cleaned up.
;  238          1141  1 !
;  239          1142  1 ! FORMAL PARAMETERS:
;  240          1143  1 !
;  241          1144  1 !       EXIT_REASON.rl.r        not used
;  242          1145  1 !
;  243          1146  1 ! IMPLICIT INPUTS:
;  244          1147  1 !
;  245          1148  1 !       NONE
;  246          1149  1 !
;  247          1150  1 ! IMPLICIT OUTPUTS:
;  248          1151  1 !
;  249          1152  1 !       BAS$$L_XIT_LOCK is zeroed.
;  250          1153  1 !
;  251          1154  1 ! ROUTINE VALUE:
;  252          1155  1 ! COMPLETION CODES:
;  253          1156  1 !
;  254          1157  1 !       NONE
;  255          1158  1 !
;  256          1159  1 ! SIDE EFFECTS:
;  257          1160  1 !
;  258          1161  1 !       Closes all files by calling BAS$$CLOSE_ALL.
;  259          1162  1 !--
;  260          1163  1
;  261          1164  2     BEGIN
;  262          1165  2     BAS$$L_XIT_LOCK = 0;                          ! Clear exit handler interlock
;  263          1166  2     BAS$$CLOSE_ALL ();
;  264          1167  1     END;                                          ! of routine EXIT_HANDLER


                                        0000 00000 EXIT_HANDLER:
                                                   .WORD   Save nothing                    ; 1129
                            00000000'   EF  D4 00002       CLRL    BAS$$L_XIT_LOCK         ; 1165
                    0000V   CF          00  FB 00008       CALLS   #0, BAS$$CLOSE_ALL      ; 1166
                                            04 0000D       RET                             ; 1167

; Routine Size:  14 bytes,    Routine Base:  _BAS$CODE + 005A

;  265          1168  1
```

```
;  267          1169  1 GLOBAL ROUTINE BAS$$CLOSE_ALL (                       ! Close all files
;  268          1170  1         PARENT_IFI                                    ! Optional IFI to look for
;  269          1171  1     ) : NOVALUE =
;  270          1172  1
;  271          1173  1 !++
;  272          1174  1 ! FUNCTIONAL DESCRIPTION:
;  273          1175  1 !
;  274          1176  1 !       Find every existing LUB (with a linear search through the LUB
;  275          1177  1 !       table).  For each LUB, if the file is open, purge its I/O
;  276          1178  1 !       buffers and close it.  If the file has been marked for PRINT
;  277          1179  1 !       or DELETE, this will cause proper disposition of the file.
;  278          1180  1 !       RMS will close all open files at image exit, but it doesn't know
;  279          1181  1 !       about the above two DISPOSE conditions.  We couldn't set them at
;  280          1182  1 !       OPEN time, since the user is allowed to specify a different
;  281          1183  1 !       DISPOSE option at close time (with the CLOSE statement).
;  282          1184  1 !       Note that BASIC does not yet have CLOSE options, so this code is
;  283          1185  1 !       a provision for the future.
;  284          1186  1 !
;  285          1187  1 ! FORMAL PARAMETERS:
;  286          1188  1 !
;  287          1189  1 !       PARENT_IFI.rl.v If present, close all files with M_STREAM set
;  288          1190  1 !                      and this IFI.  This is used by CLOSE when closing
;  289          1191  1 !                      a file which has multiple streams.  The calls to
;  290          1192  1 !                      OTS$$CLOSE_FILE will actually result in $DISCONNECTs.
;  291          1193  1 !
;  292          1194  1 ! IMPLICIT INPUTS:
;  293          1195  1 !
;  294          1196  1 !       NONE
;  295          1197  1 !
;  296          1198  1 ! IMPLICIT OUTPUTS:
;  297          1199  1 !
;  298          1200  1 !       NONE
;  299          1201  1 !
;  300          1202  1 ! ROUTINE VALUE:
;  301          1203  1 ! COMPLETION CODES:
;  302          1204  1 !
;  303          1205  1 !       NONE
;  304          1206  1 !
;  305          1207  1 ! SIDE EFFECTS:
;  306          1208  1 !
;  307          1209  1 !       Closes all files.
;  308          1210  1 !       Signals CLOSE and DISCONNECT errors as warnings.
;  309          1211  1 !--
;  310          1212  1
;  311          1213  2     BEGIN
;  312          1214  2
;  313          1215  2     BUILTIN
;  314          1216  2         NULLPARAMETER;
;  315          1217  2
;  316          1218  2     GLOBAL REGISTER
;  317          1219  2         CCB = K_CCB_REG : REF BLOCK [, BYTE];
;  318          1220  2
;  319          1221  2     LOCAL
;  320          1222  2         FLAG,
;  321          1223  2         LUN;
;  322          1224  2
;  323          1225  2 !+
```

```
;  324          1226  2 ! Scan through all BASIC logical units, closing them.
;  325          1227  2 !-
;  326          1228  2     FLAG = 0;
;  327          1229  2
;  328          1230  2     DO
;  329          1231  3         BEGIN
;  330          1232  3 !+
;  331          1233  3 ! Get the next logical unit number.
;  332          1234  3 !-
;  333          1235  3         BAS$$NEXT_LUN (FLAG, LUN);
;  334          1236  3
;  335          1237  4         IF (.FLAG NEQ 0)
;  336          1238  3         THEN
;  337          1239  4             BEGIN
;  338          1240  4 !+
;  339          1241  4 ! LUN is the next logical unit number.  If the file it represents is
;  340          1242  4 ! open try to close it.
;  341          1243  4 !-
;  342          1244  4             BAS$$CB_PUSH (.LUN, LUB$K_ILUN_MIN);
;  343          1245  4
;  344          1246  5             IF (.CCB [LUB$V_OPENED])
;  345          1247  4             THEN
;  346          1248  5                 BEGIN
;  347          1249  5
;  348          1250  6                 IF (NULLPARAMETER (1))
;  349          1251  5                 THEN
;  350          1252  6                     BEGIN
;  351          1253  6
;  352          1254  7                     IF (.CCB [LUB$V_M_STR_C])
;  353          1255  6                     THEN
;  354          1256  7                         BEGIN
;  355          1257  7 !+
;  356          1258  7 ! Close all the sons of the mother before closing the mother, i.e. if Y and
;  357          1259  7 ! Z are connected to X, and we are closing X, then we must close Y and Z and
;  358          1260  7 ! then close X.
;  359          1261  7 !-
;  360          1262  7                         BAS$$CLOSE_ALL (.CCB [LUB$W_IFI]);
;  361          1263  7                         TRY_TO_CLOSE ();
;  362          1264  7                         END
;  363          1265  6                     ELSE
;  364          1266  6                         TRY_TO_CLOSE ();
;  365          1267  6
;  366          1268  6                     END
;  367          1269  5                 ELSE
;  368          1270  5 !+
;  369          1271  5 ! Do the close (actually disconnect) only if
;  370          1272  5 ! the IFI matches and this is a connect.
;  371          1273  5 !-
;  372          1274  5
;  373          1275  5                     IF (.CCB [LUB$V_M_STREAM] AND (.CCB [LUB$W_IFI] EQL .PARENT_IFI)) THEN TRY_TO_CLOSE ();
;  374          1276  4                 END;
;  375          1277  4
;  376          1278  4             BAS$$CB_POP ();
;  377          1279  3             END;
;  378          1280  3
;  379          1281  3         END
;  380          1282  2     UNTIL (.FLAG EQL 0);
```

```
;  381          1283  2
;  382          1284  2      RETURN;
;  383          1285  1      END;                                    ! of routine BAS$$CLOSE_ALL



                                                 0804 00000          .ENTRY  BAS$$CLOSE_ALL, Save R2,R11                     ; 1169
                                          5E                 08  C2 00002  SUBL2  #8, SP                                      ;
                                                       04    AE  D4 00005  CLRL   FLAG                                       ; 1228
                                                             5E  DD 00008 1$:  PUSHL  SP                                     ; 1235
                                                       08    AE  9F 0000A  PUSHAB FLAG                                       ;
                                   00000000G           00    02  FB 0000D  CALLS  #2, BAS$$NEXT_LUN                          ;
                                                       04    AE  D5 00014  TSTL   FLAG                                       ; 1237
                                                             41  13 00017  BEQL   6$                                         ;
                                          50                 08  CE 00019  MNEGL  #8, R0                                     ; 1244
                                          52                 6E  D0 0001C  MOVL   LUN, R2                                    ;
                                             00000000G       00  16 0001F  JSB    BAS$$CB_PUSH                               ;
                                          2B           FC    AB  E9 00025  BLBC   -4(CCB), 5$                                ; 1246
                                                             6C  95 00029  TSTB   (AP)                                       ; 1250
                                                             05  13 0002B  BEQL   2$                                         ;
                                                       04    AC  D5 0002D  TSTL   4(AP)                                      ;
                                                             0F  12 00030  BNEQ   3$                                         ;
                                18        FF       AB        03  E1 00032 2$:  BBC    #3, -1(CCB), 4$                      ; 1254
                                          7E       D0  AB    3C  00037  MOVZWL -48(CCB), -(SP)                               ; 1262
                                          C1       AF        01  FB 0003B  CALLS  #1, BAS$$CLOSE_ALL                         ;
                                                             0E  11 0003F  BRB    4$                                         ; 1266
                                0E        FF       AB        02  E1 00041 3$:  BBC    #2, -1(CCB), 5$                      ; 1275
    04    AC        D0        AB            10    00  ED 00046  CMPZV  #0, #16, -48(CCB), PARENT_IFI                       ;
                                                             05  12 0004D  BNEQ   5$                                         ;
                             0000V        CF           00    FB 0004F 4$:  CALLS  #0, TRY_TO_CLOSE                         ;
                                   00000000G           00    16 00054 5$:  JSB    BAS$$CB_POP                               ; 1278
                                                 04    AE  D5 0005A 6$:  TSTL   FLAG                                       ; 1282
                                                             A9  12 0005D  BNEQ   1$                                         ;
                                                             04  0005F  RET                                                  ; 1285

; Routine Size:  96 bytes,    Routine Base:  _BAS$CODE + 0068

;  384          1286  1
```

```
;  386          1287  1 ROUTINE TRY_TO_CLOSE                          ! Call OTS$$CLOSE_FILE with errors as warnings
;  387          1288  1     : CALL_CCB NOVALUE =
;  388          1289  1
;  389          1290  1 !++
;  390          1291  1 ! FUNCTIONAL DESCRIPTION:
;  391          1292  1 !
;  392          1293  1 !       RMS CLOSE a file (by calling OTS$$CLOSE_FILE) but signal errors as warnings, to
;  393          1294  1 !       avoid losing control.
;  394          1295  1 !
;  395          1296  1 ! FORMAL PARAMETERS:
;  396          1297  1 !
;  397          1298  1 !       NONE
;  398          1299  1 !
;  399          1300  1 ! IMPLICIT INPUTS:
;  400          1301  1 !
;  401          1302  1 !       CCB             Pointer to the LUB/ISB/RAB of the file to CLOSE.
;  402          1303  1 !
;  403          1304  1 ! IMPLICIT OUTPUTS:
;  404          1305  1 !
;  405          1306  1 !       NONE
;  406          1307  1 !
;  407          1308  1 ! ROUTINE VALUE:
;  408          1309  1 ! COMPLETION CODES:
;  409          1310  1 !
;  410          1311  1 !       NONE
;  411          1312  1 !
;  412          1313  1 ! SIDE EFFECTS:
;  413          1314  1 !
;  414          1315  1 !       RMS CLOSEs the file.
;  415          1316  1 !       Signals CLOSE errors as warnings.
;  416          1317  1 !--
;  417          1318  1
;  418          1319  2     BEGIN
;  419          1320  2
;  420          1321  2     EXTERNAL REGISTER
;  421          1322  2         CCB : REF BLOCK [, BYTE];
;  422          1323  2
;  423          1324  2     ENABLE
;  424          1325  2         CLOSE_HANDLER ();
;  425          1326  2
;  426          1327  2 !+
;  427          1328  2 ! Write output buffers, then RMS CLOSE the file.
;  428          1329  2 !-
;  429          1330  2     BAS$$PUR_IO_CLO ();
;  430          1331  2
;  431          1332  2     IF ( NOT OTS$$CLOSE_FILE ()) THEN BAS$$SIGNAL_IO (BAS$K_IOERR_REC);
;  432          1333  2
;  433          1334  2     RETURN;
;  434          1335  1     END;


                                0000 00000 TRY_TO_CLOSE:
                                           .WORD   Save nothing                    ; 1287
                        6D  001D   CF  DE 00002    MOVAL   2$, (FP)                ; 1319
```

```
                00000000G  00          00  FB 00007          CALLS   #0, BAS$$PUR_IO_CLO                      ; 1330
                00000000G  00          00  FB 0000E          CALLS   #0, OTS$$CLOSE_FILE                     ; 1332
                           0A          50  E8 00015          BLBS    R0, 1$                                  ;
                           7E          01  CE 00018          MNEGL   #1, -(SP)                               ;
                00000000G  00          01  FB 0001B          CALLS   #1, BAS$$SIGNAL_IO                      ;
                                          04 00022 1$:       RET                                             ; 1335
                                    0000  00023 2$:       .WORD   Save nothing                            ; 1319
                                       7E  D4 00025          CLRL    -(SP)                                   ;
                                       5E  DD 00027          PUSHL   SP                                      ;
                           7E          04  AC  7D 00029      MOVQ    4(AP), -(SP)                            ;
                     0000V CF              03  FB 0002D      CALLS   #3, CLOSE_HANDLER                       ;
                                             04 00032       RET                                             ;

; Routine Size:  51 bytes,    Routine Base:  _BAS$CODE + 00C8
```

```
;  436          1336  1 ROUTINE CLOSE_HANDLER (                       ! Handle an error from CLOSE_ALL
;  437          1337  1         SIG,                                   ! Signal vector
;  438          1338  1         MECH,                                  ! Mechanism vector
;  439          1339  1         ENBL                                   ! Enable vector
;  440          1340  1     ) =
;  441          1341  1
;  442          1342  1 !++
;  443          1343  1 ! FUNCTIONAL DESCRIPTION:
;  444          1344  1 !
;  445          1345  1 !       If we get an error trying to close a file in CLOSE_ALL, convert the severity
;  446          1346  1 !       of the error to WARNING (if it is ERROR or SEVERE ERROR) so that we will not
;  447          1347  1 !       lose control.  It is important not to lose control so that we can try (at least)
;  448          1348  1 !       to close all the files.
;  449          1349  1 !
;  450          1350  1 ! FORMAL PARAMETERS:
;  451          1351  1 !
;  452          1352  1 !       SIG.rl.a        A counted vector of parameters to LIB$SIGNAL/STOP
;  453          1353  1 !       MECH.rl.a       A counted vector of info from CHF
;  454          1354  1 !       ENBL.ra.a       A counted vector of ENABLE argument addresses.
;  455          1355  1 !
;  456          1356  1 ! IMPLICIT INPUTS:
;  457          1357  1 !
;  458          1358  1 !       NONE
;  459          1359  1 !
;  460          1360  1 ! IMPLICIT OUTPUTS:
;  461          1361  1 !
;  462          1362  1 !       NONE
;  463          1363  1 !
;  464          1364  1 ! COMPLETION CODES:
;  465          1365  1 !
;  466          1366  1 !       Always SS$_RESIGNAL, which is ignored when unwinding.
;  467          1367  1 !
;  468          1368  1 ! SIDE EFFECTS:
;  469          1369  1 !
;  470          1370  1 !       Reduces the severity of the error to WARNING.
;  471          1371  1 !
;  472          1372  1 !--
;  473          1373  1
;  474          1374  2     BEGIN
;  475          1375  2
;  476          1376  2     MAP
;  477          1377  2         SIG : REF VECTOR,
;  478          1378  2         MECH : REF VECTOR,
;  479          1379  2         ENBL : REF VECTOR;
;  480          1380  2
;  481          1381  2     LOCAL
;  482          1382  2         COND_VALUE : BLOCK [4, BYTE];
;  483          1383  2
;  484          1384  2     COND_VALUE = .SIG [1];
;  485          1385  2 !+
;  486          1386  2 ! If the severity is ERROR or SEVERE ERROR, convert it to WARNING.
;  487          1387  2 !-
;  488          1388  2
;  489          1389  2     SELECTONE .COND_VALUE [STS$V_SEVERITY] OF
;  490          1390  2         SET
;  491          1391  2
;  492          1392  2         [STS$K_ERROR, STS$K_SEVERE] :
```

```
;  493          1393  3            BEGIN
;  494          1394  3            COND_VALUE [STS$V_SEVERITY] = STS$K_WARNING;
;  495          1395  3            SIG [1] = .COND_VALUE;
;  496          1396  2            END;
;  497          1397  2
;  498          1398  2        [OTHERWISE] :
;  499          1399  3            BEGIN
;  500          1400  3            0
;  501          1401  2            END;
;  502          1402  2        TES;
;  503          1403  2
;  504          1404  2    RETURN (SS$_RESIGNAL);
;  505          1405  1    END;                                    ! end of CLOSE_HANDLER



                                    0000 00000 CLOSE_HANDLER:
                                               .WORD   Save nothing                    ; 1336
                          50     04   AC  D0 00002          MOVL    SIG, R0            ; 1384
                          51     04   A0  D0 00006          MOVL    4(R0), COND_VALUE  ;
        02          51        03        00  ED 0000A          CMPZV   #0, #3, COND_VALUE, #2  ; 1392
                                         07  13 0000F          BEQL    1$             ;
        04          51        03        00  ED 00011          CMPZV   #0, #3, COND_VALUE, #4  ;
                                         07  12 00016          BNEQ    2$             ;
                          51             07  8A 00018 1$:     BICB2   #7, COND_VALUE     ; 1394
                    04    A0             51  D0 0001B          MOVL    COND_VALUE, 4(R0)  ; 1395
                          50     0918    8F  3C 0001F 2$:     MOVZWL  #2328, R0          ; 1404
                                         04  00024          RET                        ; 1405

; Routine Size:  37 bytes,    Routine Base:  _BAS$CODE + 00FB

;  506          1406  1 END                                    ! End of BAS$$EXIT_HANDL module
;  507          1407  1
;  508          1408  0 ELUDOM



;                                PSECT SUMMARY
;
;      Name                    Bytes                           Attributes
;
; _BAS$DATA                      24  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
; _BAS$CODE                     288  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        10         0       581          00:01.2
;
;                                       COMMAND QUALIFIERS
;
;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS$:BASEXITHA/OBJ=OBJ$:BASEXITHA MSRC$:BASEXITHA/UPDATE=(ENH$:BASEXITHA
;       )

; Size:         288 code + 24 data bytes
; Run Time:        00:12.0
; Elapsed Time:    00:28.2
; Lines/CPU Min:    7028
; Lexemes/CPU-Min: 31202
; Memory Used:  115 pages
; Compilation Complete
```

BASFREE
LIS

BASEXITHA
LIS

BASFETCHD
LIS

BASFORINI
LIS

BASGETRFA
LIS

BASFETCHA
LIS

BASGET
LIS

BASFSP
LIS

BASFIND
LIS

BASFORMAT
LIS

BASHANDLE
LIS