```
                                                       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**BASIOEND

```
BBBBBBBB      AAAAAA      SSSSSSSS      IIIIII        OOOOOO      EEEEEEEEEE  NN      NN  DDDDDDDD
BBBBBBBB      AAAAAA      SSSSSSSS      IIIIII        OOOOOO      EEEEEEEEEE  NN      NN  DDDDDDDD
BB      BB  AA      AA  SS                II        OO      OO  EE          NN      NN  DD      DD
BB      BB  AA      AA  SS                II        OO      OO  EE          NN      NN  DD      DD
BB      BB  AA      AA  SS                II        OO      OO  EE          NNNN    NN  DD      DD
BB      BB  AA      AA  SS                II        OO      OO  EE          NNNN    NN  DD      DD
BBBBBBBB    AA      AA    SSSSSS          II        OO      OO  EEEEEEEE    NN  NN  NN  DD      DD
BBBBBBBB    AA      AA    SSSSSS          II        OO      OO  EEEEEEEE    NN  NN  NN  DD      DD
BB      BB  AAAAAAAAAA          SS        II        OO      OO  EE          NN    NNNN  DD      DD
BB      BB  AAAAAAAAAA          SS        II        OO      OO  EE          NN    NNNN  DD      DD
BB      BB  AA      AA          SS        II        OO      OO  EE          NN      NN  DD      DD      ....
BB      BB  AA      AA          SS        II        OO      OO  EE          NN      NN  DD      DD      ....
BBBBBBBB    AA      AA  SSSSSSSS        IIIIII        OOOOOO      EEEEEEEEEE  NN      NN  DDDDDDDD        ....
BBBBBBBB    AA      AA  SSSSSSSS        IIIIII        OOOOOO      EEEEEEEEEE  NN      NN  DDDDDDDD        ....

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$IO_END (                           ! BASIC End I/O statement
0                 IDENT = '1-028'               ! File: BASIOEND.B32 Edit:MDL1028
0                 ) =
1 BEGIN
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1 !
1
1 !++
1 ! FACILITY: BASIC Support Library - user callable
1 !
1 ! ABSTRACT:
1 !
1 !       This module terminates a BASIC I/O statement, writes
1 !       last record if output, and pops up the I/O system to
1 !       a previously active I/O statement if any.
1 !
1 ! ENVIRONMENT: User access mode; mixture of AST level or not
1 !
1 ! AUTHOR: Donald G. Petersen, CREATION DATE: 19-Mar-78
1 !
1 ! MODIFIED BY:
1 !
1 ! 0-14  - If ISB$B_ERR_NO is non-zero, SIGNAL its contents. JMT
1 !               14-Jan-78
1 !         Donald G. Petersen, 19-Mar-78 : VERSION 1-01
1 ! 1-01  -  original BASIC
1 ! 1-02  - debugging.  DGP 07-Jun-78
1 ! 1-03  - debug.  DGP 07-Jun-78
1 ! 1-05  - If there is a Prompt outstanding at I/O end, make it a Print.
1 !           DGP 28-Sep-78
1 ! 1-06  - Change declaration of CCB from EXTERNAL to GLOBAL
1 !           DGP 09-Nov-78
1 ! 1-07  - Change to JSB linkage.  DGP 14-Nov-78
1 ! 1-009 - Add device names to REQUIRE files and update copyright
1 !            notice.  JBS 29-NOV-78
```

```
;    58        0058  1 ! 1-010 - Change LUB$B_LUN to LUB$W_LUN.  JBS 05-DEC-78
;    59        0059  1 ! 1-011 - Change REQUIRE file names from FOR... to OTS...  JBS 06-DEC-78
;    60        0060  1 ! 1-012 - Change dispatch table references to longword.  DGP 11-Dec-78
;    61        0061  1 ! 1-013 - Change calls to FOR$$CB_POP to BAS$$CB_POP.  JBS 29-DEC-78
;    62        0062  1 ! 1-014 - Change reference to FOR$$FREE_VM to LIB$FREE_VM.  DGP 16-Jan-79
;    63        0063  1 ! 1-015 - Use 32 bit addresses for externals.  JBS 27-JAN-1979
;    64        0064  1 ! 1-016 - If ISB$W_FMT_LEN is zero, don't try to free any object
;    65        0065  1 !          time format.  JBS 12-MAR-1979
;    66        0066  1 ! 1-017 - Change PRINT_POS to longword.  DGP 19-Mar-79
;    67        0067  1 ! 1-018 - Clear ISB$W_FMT_LEN before calling CB_POP.  DGP 29-May-79
;    68        0068  1 ! 1-019 - Don't actually deallocate the format string.  DGP 30-May-79
;    69        0069  1 ! 1-020 - Use language-specific dispatah tables.  JBS 26-JUN-1979
;    70        0070  1 ! 1-021 - Use ISB symbols for dispatching.  JBS 12-JUL-1979
;    71        0071  1 ! 1-022 - Set up ISB$A_USER_FP.  JBS 27-JUL-1979
;    72        0072  1 ! 1-023 - Reset LUB$V_FORM_CHAR (format character pending flag) if ISB$V_PRINT_INI
;    73        0073  1 !         is still set indicating that there were no element transmitters.  DGP
;    74        0074  1 !         07-Mar-80
;    75        0075  1 ! 1-024 - Add BAS$ANSI_IO_END entry point.  PLL 30-Jul-81
;    76        0076  1 ! 1-025 - Modify BAS$ANSI_IO_END to return a status.  PLL 22-Jul-1982
;    77        0077  1 ! 1-026 - BAS$ANSI_IO_END always returns failure - fix this!  PLL 10-Aug-1982
;    78        0078  1 ! 1-027 - TOOMUCDAT should be signalled via SIGNAL_IO, not SIGNAL.
;    79        0079  1 !         MDL 22-Nov-1982
;    80        0080  1 ! 1-028 - TOOMUCDAT should only be signalled if the buffer pointer
;    81        0081  1 !         is less than the end of the buffer rather than simply not
;    82        0082  1 !         equal to the end.  MDL 30-Nov-1982
;    83        0083  1 !--
;    84        0084  1
;    85        0085  1 !<BLF/PAGE>
```

```
0086  1 !
0087  1 ! SWITCHES:
0088  1 !
0089  1
0090  1 SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
0091  1
0092  1 !
0093  1 !       LINKAGES
0094  1 !
0095  1
0096  1 REQUIRE 'RTLIN:OTSLNK';                          ! Initialize all linkages
0525  1
0526  1 !
0527  1 ! TABLE OF CONTENTS:
0528  1 !
0529  1
0530  1 FORWARD ROUTINE
0531  1     BAS$ANSI_IO_END,                             ! ANSI entry point (for INPUT)
0532  1     BAS$IO_END : NOVALUE;                        ! End I/O statement
0533  1
0534  1 !
0535  1 ! INCLUDE FILES:
0536  1 !
0537  1
0538  1 REQUIRE 'RTLML:OTSISB';                          ! I/O statement block (ISB)
0706  1
0707  1 REQUIRE 'RTLML:OTSLUB';                          ! needed only for LUB length
0847  1
0848  1 REQUIRE 'RTLIN:OTSMAC';                          ! macros
1042  1
1043  1 REQUIRE 'RTLIN:RTLPSECT';                        ! Define DECLARE_PSECTS macro
1138  1
1139  1 REQUIRE 'RTLML:BASPAR';                          ! BASIC inter-module parameters
1161  1
1162  1 LIBRARY 'RTLSTARLE';                             ! STARLET library for macros and symbols
1163  1
1164  1 !
1165  1 ! MACROS:
1166  1 !
1167  1 !       NONE
1168  1 !
1169  1 ! EQUATED SYMBOLS:
1170  1 !
1171  1 !       NONE
1172  1 !
1173  1 ! PSECT DECLARATIONS:
1174  1 !
1175  1 DECLARE_PSECTS (BAS);                            ! declare PSECTs for BAS$ facility
1176  1 !
1177  1 ! OWN STORAGE:
1178  1 !
1179  1 !       NONE
1180  1 !
1181  1 ! EXTERNAL REFERENCES:
1182  1 !
1183  1
1184  1 EXTERNAL LITERAL
```

```
    BAS$K_TOOMUCDAT,                              ! BAS Too much data in record
    OTS$_FATINTERR;                               ! OTS Fatal Internal error

EXTERNAL
    OTS$$A_CUR_LUB : ADDRESSING_MODE (GENERAL), ! Adr of current LUB/ISB/RAB
!+
! Array of user data formatter (UDF) level of abstraction.
!-
    BAS$$AA_UDF_PR9 : VECTOR;

EXTERNAL ROUTINE
    BAS$PRINT,                                   ! BASIC Print initialize
    BAS$OUT_T_DX_S,                              ! BASIC output text element transmitter
    BAS$OUT_T_DX_C,                              ! BASIC output text element transmitter
    BAS$OUT_T_DX_B,                              ! BASIC output text element transmitter
    LIB$FREE_VM,                                 ! Return dynamically allocated virtual memory
    BAS$$CB_POP : JSB_CB_POP NOVALUE,            ! Pop entire I/O system back to previous LUB/ISB/RAB
    LIB$STOP : NOVALUE,                          ! Signal OTS errors
    BAS$$SIGNAL_IO : NOVALUE;                    ! Signal an error with a small error number
```

```
1 GLOBAL ROUTINE BAS$ANSI_IO_END   =        .
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This entr, point is used to implement Minimal ANSI INPUT only.
1 !       If the user entered more data than requested, signal a
1 !       warning message.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       OTS$$A_CUR_LUB          current I/O control block
1 !       ISB$B_STTM_TYPE         I/O statement type code
1 !       LUB$A_BUF_PTR           addr of next byte in buffer
1 !       LUB$A_BUF_END           addr+1 of last byte in buffer
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 !
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !
1 !       Signals 'too much data in record' if the user enters more data
1 !       than requested by the INPUT statement.
1 !
1 !--
1
2    BEGIN
2
2    GLOBAL REGISTER
2        CCB = K_CCB_REG : REF BLOCK [,BYTE];    ! current control block
2
2    CCB = .OTS$$A_CUR_LUB;
2
2 !+
2 ! Don't do anything if this isn't an INPUT statement.
2 !-
2
3    IF (.CCB [ISB$B_STTM_TYPE] EQL ISB$K_ST_TY_INP)
2        THEN
2            !+
2            ! ANSI semantics demand that the INPUT statement be
2            ! restarted from the beginning if any error occurs.
2            ! Most errors have already been detected by now -
2            ! BAS$$RESTART_IO is called in those cases.  Here it
2            ! doesn't make sense for the RTL to restart the
2            ! statement, since we are at the end rather than in the
2            ! middle.  So just return a status to the compiler and
2            ! let Basic re-execute its INPUT calls.
```

```
;  222          1262  2           !-
;  223          1263  3           IF (.CCB [LUB$A_BUF_PTR] LSSA .CCB [LUB$A_BUF_END])
;  224          1264  2               THEN
;  225          1265  3                   BEGIN
;  226          1266  3                   BAS$$SIGNAL_IO (BAS$K_TOOMUCDAT);
;  227          1267  3                   RETURN 1;
;  228          1268  2                   END;
;  229          1269  2
;  230          1270  2 !+
;  231          1271  2 ! ANSI INPUT processing is the same as all other I/O statements
;  232          1272  2 ! from this point on.
;  233          1273  2 !-
;  234          1274  2
;  235          1275  2     BAS$IO_END ();
;  236          1276  2
;  237          1277  2     RETURN 1;
;  238          1278  2
;  239          1279  1     END;


                                                      .TITLE  BAS$IO_END
                                                      .IDENT  \1-028\

                                                      .EXTRN  BAS$K_TOOMUCDAT
                                                      .EXTRN  OTS$_FATINTERR, OTS$$A_CUR_LUB
                                                      .EXTRN  BAS$$AA_UDF_PR9
                                                      .EXTRN  BAS$PRINT, BAS$OUT_T_DX_S
                                                      .EXTRN  BAS$OUT_T_DX_C, BAS$OUT_T_DX_B
                                                      .EXTRN  LIB$FREE_VM, BAS$$CB_POP
                                                      .EXTRN  LIB$STOP, BAS$$SIGNAL_IO

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

                                    0800 00000        .ENTRY  BAS$ANSI_IO_END, Save R11              ; 1205
                  5B 00000000G  00   D0 00002         MOVL    OTS$$A_CUR_LUB, CCB                    ; 1245
                  1E      FF71  CB   91 00009         CMPB    -143(CCB), #30                         ; 1251
                                16   12 0000E         BNEQ    1$                                     ;
           B4     AB        B0  AB   D1 00010         CMPL    -80(CCB), -76(CCB)                     ; 1263
                                0F   1E 00015         BGEQU   1$                                     ;
                     00000000G  8F   DD 00017         PUSHL   #BAS$K_TOOMUCDAT                       ; 1266
          00000000G  00         01   FB 0001D         CALLS   #1, BAS$$SIGNAL_IO                     ;
                                05   11 00024         BRB     2$                                     ; 1267
             0000V   CF         00   FB 00026 1$:     CALLS   #0, BAS$IO_END                         ; 1275
                     50         01   D0 0002B 2$:     MOVL    #1, R0                                 ; 1277
                                     04 0002E         RET                                            ; 1279

; Routine Size:  47 bytes,    Routine Base:  _BAS$CODE + 0000
```

```
1 GLOBAL ROUTINE BAS$IO_END : NOVALUE =           !
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Complete the processing of a BASIC I/O statement.  Any prompt
1 !       which has not been shown on the terminal (because it was not
1 !       followed by an input element) is turned into a PRINT.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       OTS$$A_CUR_LUB          current I/O control block
1 !       ISB$V_PRINT_INI         a Print statement was initialized
1 !       ISB$B_STTM_TYPE         I/O statement type code - index to
1 !                               dispatch table entry.
1 !       FOR$A_UDF_PR1           Array of user data formatters
1 !                               (UDF level of abstraction).
1 !       ISB$W_FMT_LEN           No. of char. allocated to object-time format or 0
1 !       ISB$A_FMT_BEG           Adr. of dynamically allocated object-time
1 !       ISB$B_ERR_NO            Last continuable error to occur in the state-
1 !                               ment or 0.  SIGNAL if non-zero!
1 !                               format array or 0 if none.
1 !       LUB$V_TERM_DEV          Indicates that the current device is a terminal.
1 !       LUB$L_PRINT_POS         Current cursor position.
1 !       ISB$V_P_FORM_CH         The format character that followed the last prompt
1 !       RAB$B_PSZ               Prompt buffer size
1 !       RAB$L_PBF               Address of the Prompt buffer
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       ISB$W_FMT_LEN           Set to 0
1 !       ISB$A_FMT_BEG           Set to 0
1 !       LUB$V_FORM_CHAR         flag indicating a format character
1 !       RAB$B_PSZ               Prompt buffer size
1 !
1 ! ROUTINE VALUE:
1 !
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !
1 !--
1
2     BEGIN
2
2     GLOBAL REGISTER
2         CCB = K_CCB_REG : REF BLOCK [, BYTE];   ! current control block
2
2     CCB = .OTS$$A_CUR_LUB;
2 !+
2 ! If the print initialized flag is still set then there were no element transmitters
```

```
2 ! and the format flag ought to be turned off before doing the PUT.
2 !-
3     IF .CCB [ISB$V_PRINT_INI] AND (.CCB [ISB$B_STTM_TYPE] EQL ISB$K_ST_TY_PRI)
2     THEN
2         CCB [LUB$V_FORM_CHAR] = 0;
2 !+
2 ! Call appropriate UDF termination routine
2 !-
2     JSB_UDF9 (BAS$$AA_UDF_PR9 + .BAS$$AA_UDF_PR9 [.CCB [ISB$B_STTM_TYPE] - ISB$K_BASSTTYLO + 1]);
2 !+
2 ! If this statement has an object-time format array allocated,
2 ! set the length and address fields back to zero so CB_POP works correctly.
2 !-
2
3     IF (.CCB [ISB$W_FMT_LEN] NEQ 0)
2     THEN
3         BEGIN
3         CCB [ISB$W_FMT_LEN] = 0;
3         CCB [ISB$A_FMT_BEG] = 0;
2         END;
2
2 !+
2 ! Check to see if there is an outstanding Prompt.  If there is and this
2 ! is a terminal device, this means that
2 ! an Input with a Prompt and no element transmitter was just processed.
2 ! Do a PRINT of the prompt buffer.  This is a case of recursive I/O.
2 !-
3     IF ((.CCB [RAB$B_PSZ] NEQU 0) AND .CCB [LUB$V_TERM_DEV])
2     THEN
3         BEGIN
3
3         LOCAL
3             T_CCB,                              ! temp for CCB-needed because CCB is a
3                                                 ! REF BLOCK
3             T_UNIT_NO,                          ! Unit on which the Prompt is pending
3             T_FORM_CHAR,                        ! temporary format char. from Prompt
3             T_DESC : BLOCK [8, BYTE],           ! temporary desc. for Print string
3             T_PRINT_POS;                        ! temporary storage for print position
3
4         T_UNIT_NO = (IF .CCB [LUB$W_LUN] LSS 0 THEN 0   ! Unit 0 which is -1 or -2 internally
3         ELSE .CCB [LUB$W_LUN]);
3         T_DESC [DSC$W_LENGTH] = .CCB [RAB$B_PSZ];
3         T_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
3         T_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
3         T_DESC [DSC$A_POINTER] = .CCB [RAB$L_PBF];
3         T_FORM_CHAR = .CCB [ISB$V_P_FORM_CH];
3         T_PRINT_POS = .CCB [LUB$L_PRINT_POS] - .CCB [RAB$B_PSZ];
3         CCB [RAB$B_PSZ] = 0;
3 !+
3 ! Initialize the Print of the outstanding Prompt.
3 !-
3         BAS$PRINT (.T_UNIT_NO);
3         T_CCB = .OTS$$A_CUR_LUB;
4         BEGIN
4
4         BUILTIN
```

```
                FP;

            LOCAL
                FMP : REF BLOCK [, BYTE];

            MAP
                T_CCB : REF BLOCK [, BYTE];

            FMP = .FP;
            T_CCB [ISB$A_USER_FP] = .FMP [SF$L_SAVE_FP];
            T_CCB [LUB$L_PRINT_POS] = .T_PRINT_POS;
            END;

            CASE .T_FORM_CHAR FROM BAS$K_SEMI_FORM TO BAS$K_NO_FORM OF
                SET

                [BAS$K_SEMI_FORM, BAS$K_COMMA_FOR] :
!+
! The dangling Prompt ended in a semicolon or a comma format char.
! Note that all processing associated with comma format character
! has already been done by the Prompt handler so we will make this
! look like a semicolon format character.
!-
                    BAS$OUT_T_DX_S (T_DESC);

                [BAS$K_NO_FORM] :
!+
! Prompt ended with no format character.
! Carriage control for Prompts is contained explicitly in the Prompt
! buffer.  This Print will now be done using VFC so we must subtract
! two from the length for the carriage control already in the buffer.
!-
                    BEGIN
                    T_DESC [DSC$W_LENGTH] = .T_DESC [DSC$W_LENGTH] - 2;
                    BAS$OUT_T_DX_B (T_DESC);
                    END;
                TES;

            BAS$IO_END ();
            END
        ELSE
!+
! Otherwise, just discard any prompt that may be left.  Prompting is
! not defined on non-terminal devices, anyway.
!-
            CCB [RAB$B_PSZ] = 0;

!+
! Indicate that we are done with this I/O statement.  If we are the last
! user of this LUB, it will be deallocated.  If we are doing recursive
! I/O, the I/O system is restored to the unit we interrupted.
! Clear ISB$W_FMT_LEN so that CB_POP doesn't try to deallocate the format
! string.
!-
        CCB [ISB$W_FMT_LEN] = 0;
        BAS$$CB_POP ();
        RETURN;
```

```
;  412         1451  1     END;                                    ! End of routine



                                                        083C 00000        .ENTRY  BAS$IO_END, Save R2,R3,R4,R5,R11             ; 1280
                                        55 00000000G  00  9E 00002        MOVAB   OTS$$A_CUR_LUB, R5                           ;
                                        5E            08  C2 00009        SUBL2   #8, SP                                       ;
                                        5B            65  D0 0000C        MOVL    OTS$$A_CUR_LUB, CCB                          ; 1334
                         0B        97   AB            03  E1 0000F        BBC     #3, -105(CCB), 1$                            ; 1339
                                        1B     FF71   CB  91 00014        CMPB    -143(CCB), #27                               ;
                                                      04  12 00019        BNEQ    1$                                           ;
                                   FE   AB            04  8A 0001B        BICB2   #4, -2(CCB)                                  ; 1341
                                        50     FF71   CB  9A 0001F 1$:    MOVZBL  -143(CCB), R0                                ; 1345
                                        50 00000000G0040  D0 00024        MOVL    BAS$$AA_UDF_PR9-104[R0], R0                  ;
                                           00000000G0040  16 0002C        JSB     BAS$$AA_UDF_PR9[R0]                          ;
                                        54     FF72   CB  9E 00033        MOVAB   -142(CCB), R4                                ; 1351
                                                      64  B5 00038        TSTW    (R4)                                         ;
                                                      06  13 0003A        BEQL    2$                                           ;
                                                      64  B4 0003C        CLRW    (R4)                                         ; 1354
                                               FF7C   CB  D4 0003E        CLRL    -132(CCB)                                    ; 1355
                                        51       34   AB  9A 00042 2$:    MOVZBL  52(CCB), R1                                  ; 1365
                                                      6F  13 00046        BEQL    9$                                           ;
                         6A        FE   AB            05  E1 00048        BBC     #5, -2(CCB), 9$                              ;
                                                 C6   AB  B5 0004D        TSTW    -58(CCB)                                     ; 1377
                                                      04  18 00050        BGEQ    3$                                           ;
                                                      50  D4 00052        CLRL    T_UNIT_NO                                    ;
                                                      04  11 00054        BRB     4$                                           ;
                                        50       C6   AB  32 00056 3$:    CVTWL   -58(CCB), T_UNIT_NO                          ; 1378
                                        6E            51  B0 0005A 4$:    MOVW    R1, T_DESC                                   ; 1379
                         02             AE     010E   8F  B0 0005D        MOVW    #270, T_DESC+2                               ; 1380
                         04             AE       30   AB  D0 00063        MOVL    48(CCB), T_DESC+4                            ; 1382
           53       96   AB             02            00  EF 00068        EXTZV   #0, #2, -106(CCB), T_FORM_CHAR               ; 1383
                         52        C8   AB            51  C3 0006E        SUBL3   R1, -56(CCB), T_PRINT_POS                    ; 1384
                                                 34   AB  94 00073        CLRB    52(CCB)                                      ; 1385
                                                      50  DD 00076        PUSHL   T_UNIT_NO                                    ; 1389
                             00000000G  00            01  FB 00078        CALLS   #1, BAS$PRINT                                ;
                                        50            65  D0 0007F        MOVL    OTS$$A_CUR_LUB, T_CCB                        ; 1390
                                        51            5D  D0 00082        MOVL    FP, FMP                                      ; 1402
                                 FF4C   C0       0C   A1  D0 00085        MOVL    12(FMP), -180(T_CCB)                         ; 1403
                                   C8   A0            52  D0 0008B        MOVL    T_PRINT_POS, -56(T_CCB)                      ; 1404
                         02             01            53  CF 0008F        CASEL   T_FORM_CHAR, #1, #2                          ; 1407
                       0011           0006          0006     00093 5$:    .WORD   6$-5$,-                                      ;
                                                                                  6$-5$,-                                      ;
                                                                                  7$-5$                                        ;
                                                      5E  DD 00099 6$:    PUSHL   SP                                           ; 1417
                             00000000G  00            01  FB 0009B        CALLS   #1, BAS$OUT_T_DX_S                           ;
                                                      0C  11 000A2        BRB     8$                                           ;
                                        6E            02  A2 000A4 7$:    SUBW2   #2, T_DESC                                   ; 1427
                                                      5E  DD 000A7        PUSHL   SP                                           ; 1428
                             00000000G  00            01  FB 000A9        CALLS   #1, BAS$OUT_T_DX_B                           ;
                                 FF4B   CF            00  FB 000B0 8$:    CALLS   #0, BAS$IO_END                               ; 1432
                                                      03  11 000B5        BRB     10$                                          ; 1365
                                                 34   AB  94 000B7 9$:    CLRB    52(CCB)                                      ; 1439
                                                      64  B4 000BA 10$:   CLRW    (R4)                                         ; 1448
                                           00000000G  00  16 000BC        JSB     BAS$$CB_POP                                  ; 1449
```

```
                                               04 000C2         RET                                        ; 1451

; Routine Size:  195 bytes,    Routine Base:  _BAS$CODE + 002F


;  413           1452  1
;  414           1453  1 END                                          !End of module BAS$IO_END
;  415           1454  1
;  416           1455  0 ELUDOM



;                                 PSECT SUMMARY
;
;      Name                   Bytes                        Attributes
;
;  _BAS$CODE                      242  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         9         0        581          00:01.1



;                                  COMMAND QUALIFIERS

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

; Size:          242 code + 0 data bytes
; Run Time:         00:12.8
; Elapsed Time:     00:28.4
; Lines/CPU Min:     6836
; Lexemes/CPU-Min: 39406
; Memory Used:  166 pages
; Compilation Complete
```

0024 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

BASINIGSC
LIS

BASINIT
LIS

BASINIDEF
LIS

BASINIDFS
LIS

BASINIGSB
LIS

BASINSTR
LIS

BASINIONE
LIS

BASLEFT
LIS

BASMARGIN
LIS

BASINIIOL
LIS

BASKILL
LIS

BASIOBEG
LIS

BASIOEND
LIS

BASMATADD
LIS

BASMAGTAP
LIS