```
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**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
```

```
;     1          0001  0 MODULE BAS$IO_END (                          ! BASIC End I/O statement
;     2          0002  0                 IDENT = '1-028'             ! File: BASIOEND.B32 Edit:MDL1028
;     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 !++
;    31          0031  1 ! FACILITY: BASIC Support Library - user callable
;    32          0032  1 !
;    33          0033  1 ! ABSTRACT:
;    34          0034  1 !
;    35          0035  1 !       This module terminates a BASIC I/O statement, writes
;    36          0036  1 !       last record if output, and pops up the I/O system to
;    37          0037  1 !       a previously active I/O statement if any.
;    38          0038  1 !
;    39          0039  1 ! ENVIRONMENT: User access mode; mixture of AST level or not
;    40          0040  1 !
;    41          0041  1 ! AUTHOR: Donald G. Petersen, CREATION DATE: 19-Mar-78
;    42          0042  1 !
;    43          0043  1 ! MODIFIED BY:
;    44          0044  1 !
;    45          0045  1 ! 0-14  - If ISB$B_ERR_NO is non-zero, SIGNAL its contents. JMT
;    46          0046  1 !                 14-Jan-78
;    47          0047  1 !       Donald G. Petersen, 19-Mar-78 : VERSION 1-01
;    48          0048  1 ! 1-01  -  original BASIC
;    49          0049  1 ! 1-02  - debugging.  DGP 07-Jun-78
;    50          0050  1 ! 1-03  - debug.  DGP 07-Jun-78
;    51          0051  1 ! 1-05  - If there is a Prompt outstanding at I/O end, make it a Print.
;    52          0052  1 !         DGP 28-Sep-78
;    53          0053  1 ! 1-06  - Change declaration of CCB from EXTERNAL to GLOBAL
;    54          0054  1 !         DGP 09-Nov-78
;    55          0055  1 ! 1-07  - Change to JSB linkage.  DGP 14-Nov-78
;    56          0056  1 ! 1-009 - Add device names to REQUIRE files and update copyright
;    57          0057  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>
```

```
;    87          0086  1 !
;    88          0087  1 ! SWITCHES:
;    89          0088  1 !
;    90          0089  1
;    91          0090  1 SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
;    92          0091  1
;    93          0092  1 !
;    94          0093  1 !       LINKAGES
;    95          0094  1 !
;    96          0095  1
;    97          0096  1 REQUIRE 'RTLIN:OTSLNK';                         ! Initialize all linkages
;    98          0525  1
;    99          0526  1 !
;   100          0527  1 ! TABLE OF CONTENTS:
;   101          0528  1 !
;   102          0529  1
;   103          0530  1 FORWARD ROUTINE
;   104          0531  1     BAS$ANSI_IO_END,                            ! ANSI entry point (for INPUT)
;   105          0532  1     BAS$IO_END : NOVALUE;                       ! End I/O statement
;   106          0533  1
;   107          0534  1 !
;   108          0535  1 ! INCLUDE FILES:
;   109          0536  1 !
;   110          0537  1
;   111          0538  1 REQUIRE 'RTLML:OTSISB';                         ! I/O statement block (ISB)
;   112          0706  1
;   113          0707  1 REQUIRE 'RTLML:OTSLUB';                         ! needed only for LUB length
;   114          0847  1
;   115          0848  1 REQUIRE 'RTLIN:OTSMAC';                         ! macros
;   116          1042  1
;   117          1043  1 REQUIRE 'RTLIN:RTLPSECT';                       ! Define DECLARE_PSECTS macro
;   118          1138  1
;   119          1139  1 REQUIRE 'RTLML:BASPAR';                         ! BASIC inter-module parameters
;   120          1161  1
;   121          1162  1 LIBRARY 'RTLSTARLE';                            ! STARLET library for macros and symbols
;   122          1163  1
;   123          1164  1 !
;   124          1165  1 ! MACROS:
;   125          1166  1 !
;   126          1167  1 !       NONE
;   127          1168  1 !
;   128          1169  1 ! EQUATED SYMBOLS:
;   129          1170  1 !
;   130          1171  1 !       NONE
;   131          1172  1 !
;   132          1173  1 ! PSECT DECLARATIONS:
;   133          1174  1 !
;   134          1175  1 DECLARE_PSECTS (BAS);                           ! declare PSECTs for BAS$ facility
;   135          1176  1 !
;   136          1177  1 ! OWN STORAGE:
;   137          1178  1 !
;   138          1179  1 !       NONE
;   139          1180  1 !
;   140          1181  1 ! EXTERNAL REFERENCES:
;   141          1182  1 !
;   142          1183  1
;   143          1184  1 EXTERNAL LITERAL
```

```
;  144          1185  1      BAS$K_TOOMUCDAT,                        ! BAS Too much data in record
;  145          1186  1      OTS$_FATINTERR;                         ! OTS Fatal Internal error
;  146          1187  1
;  147          1188  1  EXTERNAL
;  148          1189  1      OTS$$A_CUR_LUB : ADDRESSING_MODE (GENERAL), ! Adr of current LUB/ISB/RAB
;  149          1190  1  !+
;  150          1191  1  ! Array of user data formatter (UDF) level of abstraction.
;  151          1192  1  !-
;  152          1193  1      BAS$$AA_UDF_PR9 : VECTOR;
;  153          1194  1
;  154          1195  1  EXTERNAL ROUTINE
;  155          1196  1      BAS$PRINT,                                  ! BASIC Print initialize
;  156          1197  1      BAS$OUT_T_DX_S,                             ! BASIC output text element transmitter
;  157          1198  1      BAS$OUT_T_DX_C,                             ! BASIC output text element transmitter
;  158          1199  1      BAS$OUT_T_DX_B,                             ! BASIC output text element transmitter
;  159          1200  1      LIB$FREE_VM,                                ! Return dynamically allocated virtual memory
;  160          1201  1      BAS$$CB_POP : JSB_CB_POP NOVALUE,           ! Pop entire I/O system back to previous LUB/ISB/RAB
;  161          1202  1      LIB$STOP : NOVALUE,                         ! Signal OTS errors
;  162          1203  1      BAS$$SIGNAL_IO : NOVALUE;                   ! Signal an error with a small error number
;  163          1204  1
```

```
;  165          1205  1 GLOBAL ROUTINE BAS$ANSI_IO_END   =       !
;  166          1206  1
;  167          1207  1 !++
;  168          1208  1 ! FUNCTIONAL DESCRIPTION:
;  169          1209  1 !
;  170          1210  1 !       This entry point is used to implement Minimal ANSI INPUT only.
;  171          1211  1 !       If the user entered more data than requested, signal a
;  172          1212  1 !       warning message.
;  173          1213  1 !
;  174          1214  1 ! FORMAL PARAMETERS:
;  175          1215  1 !
;  176          1216  1 !       NONE
;  177          1217  1 !
;  178          1218  1 ! IMPLICIT INPUTS:
;  179          1219  1 !
;  180          1220  1 !       OTS$$A_CUR_LUB          current I/O control block
;  181          1221  1 !       ISB$B_STTM_TYPE         I/O statement type code
;  182          1222  1 !       LUB$A_BUF_PTR           addr of next byte in buffer
;  183          1223  1 !       LUB$A_BUF_END           addr+1 of last byte in buffer
;  184          1224  1 !
;  185          1225  1 ! IMPLICIT OUTPUTS:
;  186          1226  1 !
;  187          1227  1 !       NONE
;  188          1228  1 !
;  189          1229  1 ! ROUTINE VALUE:
;  190          1230  1 !
;  191          1231  1 !       NONE
;  192          1232  1 !
;  193          1233  1 ! SIDE EFFECTS:
;  194          1234  1 !
;  195          1235  1 !       Signals 'too much data in record' if the user enters more data
;  196          1236  1 !       than requested by the INPUT statement.
;  197          1237  1 !
;  198          1238  1 !--
;  199          1239  1
;  200          1240  2     BEGIN
;  201          1241  2
;  202          1242  2     GLOBAL REGISTER
;  203          1243  2         CCB = K_CCB_REG : REF BLOCK [,BYTE];    ! current control block
;  204          1244  2
;  205          1245  2     CCB = .OTS$$A_CUR_LUB;
;  206          1246  2
;  207          1247  2 !+
;  208          1248  2 ! Don't do anything if this isn't an INPUT statement.
;  209          1249  2 !-
;  210          1250  2
;  211          1251  3     IF (.CCB [ISB$B_STTM_TYPE] EQL ISB$K_ST_TY_INP)
;  212          1252  2         THEN
;  213          1253  2             !+
;  214          1254  2             ! ANSI semantics demand that the INPUT statement be
;  215          1255  2             ! restarted from the beginning if any error occurs.
;  216          1256  2             ! Most errors have already been detected by now -
;  217          1257  2             ! BAS$$RESTART_IO is called in those cases.  Here it
;  218          1258  2             ! doesn't make sense for the RTL to restart the
;  219          1259  2             ! statement, since we are at the end rather than in the
;  220          1260  2             ! middle.  So just return a status to the compiler and
;  221          1261  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  3                  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
```

```
;  241     1280  1 GLOBAL ROUTINE BAS$IO_END : NOVALUE =          !
;  242     1281  1
;  243     1282  1 !++
;  244     1283  1 ! FUNCTIONAL DESCRIPTION:
;  245     1284  1 !
;  246     1285  1 !       Complete the processing of a BASIC I/O statement.  Any prompt
;  247     1286  1 !       which has not been shown on the terminal (because it was not
;  248     1287  1 !       followed by an input element) is turned into a PRINT.
;  249     1288  1 !
;  250     1289  1 ! FORMAL PARAMETERS:
;  251     1290  1 !
;  252     1291  1 !       NONE
;  253     1292  1 !
;  254     1293  1 ! IMPLICIT INPUTS:
;  255     1294  1 !
;  256     1295  1 !       OTS$$A_CUR_LUB          current I/O control block
;  257     1296  1 !       ISB$V_PRINT_INI         a Print statement was initialized
;  258     1297  1 !       ISB$B_STTM_TYPE         I/O statement type code - index to
;  259     1298  1 !                               dispatch table entry.
;  260     1299  1 !       FOR$A_UDF_PR1           Array of user data formatters
;  261     1300  1 !                               (UDF level of abstraction).
;  262     1301  1 !       ISB$W_FMT_LEN           No. of char. allocated to object-time format or 0
;  263     1302  1 !       ISB$A_FMT_BEG           Adr. of dynamically allocated object-time
;  264     1303  1 !       ISB$B_ERR_NO            Last continuable error to occur in the state-
;  265     1304  1 !                               ment or 0.  SIGNAL if non-zero!
;  266     1305  1 !                               format array or 0 if none.
;  267     1306  1 !       LUB$V_TERM_DEV          Indicates that the current device is a terminal.
;  268     1307  1 !       LUB$L_PRINT_POS         Current cursor position.
;  269     1308  1 !       ISB$V_P_FORM_CH         The format character that followed the last prompt
;  270     1309  1 !       RAB$B_PSZ               Prompt buffer size
;  271     1310  1 !       RAB$L_PBF               Address of the Prompt buffer
;  272     1311  1 !
;  273     1312  1 ! IMPLICIT OUTPUTS:
;  274     1313  1 !
;  275     1314  1 !       ISB$W_FMT_LEN           Set to 0
;  276     1315  1 !       ISB$A_FMT_BEG           Set to 0
;  277     1316  1 !       LUB$V_FORM_CHAR         flag indicating a format character
;  278     1317  1 !       RAB$B_PSZ               Prompt buffer size
;  279     1318  1 !
;  280     1319  1 ! ROUTINE VALUE:
;  281     1320  1 !
;  282     1321  1 !       NONE
;  283     1322  1 !
;  284     1323  1 ! SIDE EFFECTS:
;  285     1324  1 !
;  286     1325  1 !       NONE
;  287     1326  1 !
;  288     1327  1 !--
;  289     1328  1
;  290     1329  2     BEGIN
;  291     1330  2
;  292     1331  2     GLOBAL REGISTER
;  293     1332  2         CCB = K_CCB_REG : REF BLOCK [, BYTE];   ! current control block
;  294     1333  2
;  295     1334  2     CCB = .OTS$$A_CUR_LUB;
;  296     1335  2 !+
;  297     1336  2 ! If the print initialized flag is still set then there were no element transmitters
```

```
;  298          1337  2 ! and the format flag ought to be turned off before doing the PUT.
;  299          1338  2 !-
;  300          1339  2     IF .CCB [ISB$V_PRINT_INI] AND (.CCB [ISB$B_STTM_TYPE] EQL ISB$K_ST_TY_PRI)
;  301          1340  2     THEN
;  302          1341  2         CCB [LUB$V_FORM_CHAR] = 0;
;  303          1342  2 !+
;  304          1343  2 ! Call appropriate UDF termination routine
;  305          1344  2 !-
;  306          1345  2     JSB_UDF9 (BAS$$AA_UDF_PR9 + .BAS$$AA_UDF_PR9 [.CCB [ISB$B_STTM_TYPE] - ISB$K_BASSTTYLO + 1]);
;  307          1346  2 !+
;  308          1347  2 ! If this statement has an object-time format array allocated,
;  309          1348  2 ! set the length and address fields back to zero so CB_POP works correctly.
;  310          1349  2 !-
;  311          1350  2
;  312          1351  2     IF (.CCB [ISB$W_FMT_LEN] NEQ 0)
;  313          1352  2     THEN
;  314          1353  3         BEGIN
;  315          1354  3         CCB [ISB$W_FMT_LEN] = 0;
;  316          1355  3         CCB [ISB$A_FMT_BEG] = 0;
;  317          1356  3         END;
;  318          1357  2
;  319          1358  2 !+
;  320          1359  2 ! Check to see if there is an outstanding Prompt.  If there is and this
;  321          1360  2 ! is a terminal device, this means that
;  322          1361  2 ! an Input with a Prompt and no element transmitter was just processed.
;  323          1362  2 ! Do a PRINT of the prompt buffer.  This is a case of recursive I/O.
;  324          1363  2 !-
;  325          1364  2
;  326          1365  3     IF ((.CCB [RAB$B_PSZ] NEQU 0) AND .CCB [LUB$V_TERM_DEV])
;  327          1366  2     THEN
;  328          1367  3         BEGIN
;  329          1368  3
;  330          1369  3         LOCAL
;  331          1370  3             T_CCB,                              ! temp for CCB-needed because CCB is a
;  332          1371  3                                                 ! REF BLOCK
;  333          1372  3             T_UNIT_NO,                          ! Unit on which the Prompt is pending
;  334          1373  3             T_FORM_CHAR,                        ! temporary format char. from Prompt
;  335          1374  3             T_DESC : BLOCK [8, BYTE],           ! temporary desc. for Print string
;  336          1375  3             T_PRINT_POS;                        ! temporary storage for print position
;  337          1376  3
;  338          1377  4         T_UNIT_NO = (IF .CCB [LUB$W_LUN] LSS 0 THEN 0   ! Unit 0 which is -1 or -2 internally
;  339          1378  4         ELSE .CCB [LUB$W_LUN]);
;  340          1379  3         T_DESC [DSC$W_LENGTH] = .CCB [RAB$B_PSZ];
;  341          1380  3         T_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  342          1381  3         T_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  343          1382  3         T_DESC [DSC$A_POINTER] = .CCB [RAB$L_PBF];
;  344          1383  3         T_FORM_CHAR = .CCB [ISB$V_P_FORM_CH];
;  345          1384  3         T_PRINT_POS = .CCB [LUB$L_PRINT_POS] - .CCB [RAB$B_PSZ];
;  346          1385  3         CCB [RAB$B_PSZ] = 0;
;  347          1386  3 !+
;  348          1387  3 ! Initialize the Print of the outstanding Prompt.
;  349          1388  3 !-
;  350          1389  3         BAS$PRINT (.T_UNIT_NO);
;  351          1390  3         T_CCB = .OTS$$A_CUR_LUB;
;  352          1391  4         BEGIN
;  353          1392  4
;  354          1393  4         BUILTIN
```

```
;  355          1394  4              FP;
;  356          1395  4
;  357          1396  4          LOCAL
;  358          1397  4              FMP : REF BLOCK [, BYTE];
;  359          1398  4
;  360          1399  4          MAP
;  361          1400  4              T_CCB : REF BLOCK [, BYTE];
;  362          1401  4
;  363          1402  4          FMP = .FP;
;  364          1403  4          T_CCB [ISB$A_USER_FP] = .FMP [SF$L_SAVE_FP];
;  365          1404  4          T_CCB [LUB$L_PRINT_POS] = .T_PRINT_POS;
;  366          1405  3          END;
;  367          1406  3
;  368          1407  3          CASE .T_FORM_CHAR FROM BAS$K_SEMI_FORM TO BAS$K_NO_FORM OF
;  369          1408  3              SET
;  370          1409  3
;  371          1410  3              [BAS$K_SEMI_FORM, BAS$K_COMMA_FOR] :
;  372          1411  3      !+
;  373          1412  3      ! The dangling Prompt ended in a semicolon or a comma format char.
;  374          1413  3      ! Note that all processing associated with comma format character
;  375          1414  3      ! has already been done by the Prompt handler so we will make this
;  376          1415  3      ! look like a semicolon format character.
;  377          1416  3      !-
;  378          1417  3                  BAS$OUT_T_DX_S (T_DESC);
;  379          1418  3
;  380          1419  3              [BAS$K_NO_FORM] :
;  381          1420  3      !+
;  382          1421  3      ! Prompt ended with no format character.
;  383          1422  3      ! Carriage control for Prompts is contained explicitly in the Prompt
;  384          1423  3      ! buffer.  This Print will now be done using VFC so we must subtract
;  385          1424  3      ! two from the length for the carriage control already in the buffer.
;  386          1425  3      !-
;  387          1426  4                  BEGIN
;  388          1427  4                  T_DESC [DSC$W_LENGTH] = .T_DESC [DSC$W_LENGTH] - 2;
;  389          1428  4                  BAS$OUT_T_DX_B (T_DESC);
;  390          1429  3                  END;
;  391          1430  3              TES;
;  392          1431  3
;  393          1432  3          BAS$IO_END ();
;  394          1433  3          END
;  395          1434  2      ELSE
;  396          1435  2      !+
;  397          1436  2      ! Otherwise, just discard any prompt that may be left.  Prompting is
;  398          1437  2      ! not defined on non-terminal devices, anyway.
;  399          1438  2      !-
;  400          1439  2          CCB [RAB$B_PSZ] = 0;
;  401          1440  2
;  402          1441  2      !+
;  403          1442  2      ! Indicate that we are done with this I/O statement.  If we are the last
;  404          1443  2      ! user of this LUB, it will be deallocated.  If we are doing recursive
;  405          1444  2      ! I/O, the I/O system is restored to the unit we interrupted.
;  406          1445  2      ! Clear ISB$W_FMT_LEN so that CB_POP doesn't try to deallocate the format
;  407          1446  2      ! string.
;  408          1447  2      !-
;  409          1448  2          CCB [ISB$W_FMT_LEN] = 0;
;  410          1449  2          BAS$$CB_POP ();
;  411          1450  2          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  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  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  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  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
```

BASINIGSC
LIS

BASINIT
LIS

BASINIDEF
LIS

BASINIDFS
LIS

BASINIGSB
LIS

BASINSTR
LIS

BASINIONE
LIS

BASLEFT
LIS

BASMARGIN
LIS

BASINIIOL
LIS

BASKILL
LIS

BASTOBEG
LIS

BASTOEND
LIS

BASMATADD
LIS

BASMAGTAP
LIS