```
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**BASSTR

```
BBBBBBBB      AAAAAA      SSSSSSSS    SSSSSSSS  TTTTTTTTTT  RRRRRRRR
BBBBBBBB      AAAAAA      SSSSSSSS    SSSSSSSS  TTTTTTTTTT  RRRRRRRR
BB      BB   AA    AA   SS          SS            TT      RR      RR
BB      BB   AA    AA   SS          SS            TT      RR      RR
BB      BB   AA    AA   SS          SS            TT      RR      RR
BB      BB   AA    AA   SS          SS            TT      RR      RR
BBBBBBBB     AA    AA     SSSSSS      SSSSSS      TT      RRRRRRRR
BBBBBBBB     AA    AA     SSSSSS      SSSSSS      TT      RRRRRRRR
BB      BB   AAAAAAAAAA         SS          SS    TT      RR  RR
BB      BB   AAAAAAAAAA         SS          SS    TT      RR  RR
BB      BB   AA    AA           SS          SS    TT      RR    RR      ....
BB      BB   AA    AA           SS          SS    TT      RR    RR      ....
BBBBBBBB     AA    AA   SSSSSSSS    SSSSSSSS      TT      RR      RR    ....
BBBBBBBB     AA    AA   SSSSSSSS    SSSSSSSS      TT      RR      RR    ....

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$STR (                      ! Routines to do BASIC STR$ function
;     2          0002  0                 IDENT = '1-008'       ! module BASSTR.B32 Edit: PLL1008
;     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 !++
;    32          0032  1 ! FACILITY: BASIC Support Library
;    33          0033  1 !
;    34          0034  1 ! ABSTRACT:
;    35          0035  1 !
;    36          0036  1 !       This module has entry points long, floating, double,
;    37          0037  1 !       g floating, and h floating.
;    38          0038  1 !       The double routine checks for a BASIC frame and picks
;    39          0039  1 !       up the scale factor.  Then all routines convert a number
;    40          0040  1 !       to a numeric string as it would be formatted by the BASIC print
;    41          0041  1 !       statement but without leading or trailing spaces (by a CALL to the
;    42          0042  1 !       correct BAS$ conversion routine).
;    43          0043  1 !
;    44          0044  1 ! ENVIRONMENT: User mode, AST level or not or mixed
;    45          0045  1 !
;    46          0046  1 ! AUTHOR: R. Will, CREATION DATE: 8-Mar-79
;    47          0047  1 !
;    48          0048  1 ! MODIFIED BY:
;    49          0049  1 !
;    50          0050  1 ! R. Will, 8-Mar-79: VERSION 01
;    51          0051  1 ! 01    - original
;    52          0052  1 ! 1-002 - Prefix string linkages with STR$.  JBS 04-JUN-1979
;    53          0053  1 ! 1-003 - Add BASLNK for scaling linkages.  RW  26-Jun-79
;    54          0054  1 ! 1-004 - Change to use new conversion routines.  RW 7-Jul-79
;    55          0055  1 ! 1-005 - Add longword entry point.  RW  10-Sept-79
;    56          0056  1 ! 1-006 - String cleanup, don't use $STR$ macros.  RW  30-Oct-79
;    57          0057  1 ! 1-007 - Add entry points for g & h floating.  PLL 3-Sep-81
```

```
;   58          0058  1 ! 1-008 - Add entry point for packed decimal.  PLL 19-Jan-82
;   59          0059  1 !--
;   60          0060  1 !<BLF/PAGE>
```

```
;    62          0061  1 !
;    63          0062  1 ! SWITCHES:
;    64          0063  1 !
;    65          0064  1
;    66          0065  1 SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
;    67          0066  1
;    68          0067  1 !
;    69          0068  1 ! LINKAGES: NONE
;    70          0069  1 !
;    71          0070  1
;    72          0071  1 !
;    73          0072  1 ! TABLE OF CONTENTS:
;    74          0073  1 !
;    75          0074  1
;    76          0075  1 FORWARD ROUTINE
;    77          0076  1     BAS$STR_L : NOVALUE,                    ! Find STR$ of a longword value
;    78          0077  1     BAS$STR_F : NOVALUE,                    ! Find STR$ of a floating value
;    79          0078  1     BAS$STR_D : NOVALUE,                    ! Find STR$ of a double value
;    80          0079  1     BAS$STR_G : NOVALUE,                    ! Find STR$ of a g float value
;    81          0080  1     BAS$STR_H : NOVALUE,                    ! Find STR$ of an h float value
;    82          0081  1     BAS$STR_P : NOVALUE;                    ! Find STR$ of a decimal value
;    83          0082  1
;    84          0083  1 !
;    85          0084  1 ! INCLUDE FILES:
;    86          0085  1 !
;    87          0086  1
;    88          0087  1 REQUIRE 'RTLIN:RTLPSECT';                   ! Declare PSECTs code
;    89          0182  1 REQUIRE 'RTLIN:BASLNK';                     ! Linkages for BASIC scaling
;    90          0259  1 REQUIRE 'RTLIN:BASFRAME';                   ! Define offsets in a BASIC frame
;    91          0462  1
;    92          0463  1 !
;    93          0464  1 ! MACROS: NONE
;    94          0465  1 !
;    95          0466  1
;    96          0467  1 !
;    97          0468  1 ! EQUATED SYMBOLS:
;    98          0469  1 !
;    99          0470  1
;   100          0471  1 LITERAL
;   101          0472  1     digits_in_long = 10,                    ! # of digits to display for longword
;   102          0473  1                                             !  note: float & double use the default
;   103          0474  1     strip_spaces = 1;                       ! flag for stripping spaces
;   104          0475  1
;   105          0476  1 !
;   106          0477  1 ! PSECT DECLARATIONS
;   107          0478  1 !
;   108          0479  1
;   109          0480  1 DECLARE_PSECTS (BAS);
;   110          0481  1
;   111          0482  1 !
;   112          0483  1 ! OWN STORAGE: NONE
;   113          0484  1 !
;   114          0485  1
;   115          0486  1 !
;   116          0487  1 ! EXTERNAL REFERENCES:
;   117          0488  1 !
;   118          0489  1
```

```
;  119          0490  1 EXTERNAL ROUTINE
;  120          0491  1     BAS$CVT_OUT_D_G,          ! Convert dbl to BASIC string format
;  121          0492  1     BAS$CVT_OUT_G_G,          ! Convert gfloat to BASIC string format
;  122          0493  1     BAS$CVT_OUT_H_G,          ! Convert hfloat to BASIC string format
;  123          0494  1     BAS$CVT_OUT_P_G;          ! Convert packed to BASIC string format
;  124          0495  1
;  125          0496  1 BUILTIN
;  126          0497  1     CVTLD;                    ! Convert long to double to call CVT rtn
```

```
;  128     0498  1 GLOBAL ROUTINE BAS$STR_L (                    ! convert integer to string
;  129     0499  1                             STRING,            ! Address of destination descriptor
;  130     0500  1                             VALUE) :           ! Find numeric value of this number
;  131     0501  1                             NOVALUE =
;  132     0502  1
;  133     0503  1 !++
;  134     0504  1 ! FUNCTIONAL DESCRIPTION:
;  135     0505  1 !
;  136     0506  1 !       This routine takes a longword integer and formats it as the BASIC PRINT
;  137     0507  1 !       statement would without leading and trailing spaces
;  138     0508  1 !       and gives that value to the destination string.
;  139     0509  1 !
;  140     0510  1 ! FORMAL PARAMETERS:
;  141     0511  1 !
;  142     0512  1 !       STRING.wt.dx            pointer to input string descriptor
;  143     0513  1 !       VALUE.rl.v              value of a longword number
;  144     0514  1 !
;  145     0515  1 ! IMPLICIT INPUTS:
;  146     0516  1 !
;  147     0517  1 !       NONE
;  148     0518  1 !
;  149     0519  1 ! IMPLICIT OUTPUTS:
;  150     0520  1 !
;  151     0521  1 !       NONE
;  152     0522  1 !
;  153     0523  1 ! ROUTINE VALUE:
;  154     0524  1 ! COMPLETION CODES:
;  155     0525  1 !
;  156     0526  1 !       NONE
;  157     0527  1 !
;  158     0528  1 ! SIDE EFFECTS:
;  159     0529  1 !
;  160     0530  1 !       This routine calls the conversion routine and so may signal any of its
;  161     0531  1 !       errors or have any of its side effects.  In particular, the conversion
;  162     0532  1 !       routine calls STR$ routines and so may allocate or deallocate dynamic
;  163     0533  1 !       string space, and lock a string from being written for a period.
;  164     0534  1 !
;  165     0535  1 !--
;  166     0536  1
;  167     0537  2     BEGIN
;  168     0538  2
;  169     0539  2     MAP
;  170     0540  2         STRING : REF BLOCK [8,BYTE];
;  171     0541  2
;  172     0542  2     LOCAL
;  173     0543  2         STR_LENGTH : WORD,                      ! conversion rtn returns len
;  174     0544  2         TEMP : VECTOR [2, LONG];                ! need double to pass to cnv
;  175     0545  2
;  176     0546  2     CVTLD (VALUE, TEMP [0]);                    ! make value into double
;  177     0547  2     BAS$CVT_OUT_D_G (TEMP [0],                  ! convert this value to string
;  178     0548  2                      strip_spaces,              ! set flag to strip spaces
;  179     0549  2                      STR_LENGTH,                ! return bytes needed for str
;  180     0550  2                      STRING [0,0,0,0],          ! descriptor of result string
;  181     0551  2                      0,                         ! no scale factor
;  182     0552  2                      digits_in_long);           ! # of significant digits
;  183     0553  2
;  184     0554  2     RETURN;
```

```
;  185          0555  1     END;                                    !End of BAS$STR_L


                                                                     .TITLE  BAS$STR
                                                                     .IDENT  \1-008\

                                                                     .EXTRN  BAS$CVT_OUT_D_G
                                                                     .EXTRN  BAS$CVT_OUT_G_G
                                                                     .EXTRN  BAS$CVT_OUT_H_G
                                                                     .EXTRN  BAS$CVT_OUT_P_G

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

                                          0000 00000                 .ENTRY  BAS$STR_L, Save nothing                 ; 0498
                                    5E    0000 C2 00002              SUBL2   #12, SP                                 ;
                                04  AE    0003 AC 6E 00005           CVTLD   VALUE, TEMP                             ; 0546
                                          0008 DD 0000A              PUSHL   #10                                     ; 0550
                                          000A D4 0000C              CLRL    -(SP)                                   ;
                                          000C AC DD 0000E           PUSHL   STRING                                  ;
                                          0010 AE 9F 00011           PUSHAB  STR_LENGTH                              ; 0547
                                          0013 DD 00014              PUSHL   #1                                      ; 0550
                                          0015 AE 9F 00016           PUSHAB  TEMP                                    ; 0547
                00000000G         00    0018 FB 00019              CALLS   #6, BAS$CVT_OUT_D_G                     ; 0550
                                          001F 04 00020              RET                                             ; 0555

; Routine Size:  33 bytes,    Routine Base:  _BAS$CODE + 0000
```

```
;  187     0556  1 GLOBAL ROUTINE BAS$STR_F (                    ! floating number to a string
;  188     0557  1                             STRING,            ! Address of destination descriptor
;  189     0558  1                             VALUE) :           ! Find numeric value of this string
;  190     0559  1                             NOVALUE =
;  191     0560  1
;  192     0561  1 !++
;  193     0562  1 ! FUNCTIONAL DESCRIPTION:
;  194     0563  1 !
;  195     0564  1 !       This routine takes a floating number and formats it as the BASIC PRINT
;  196     0565  1 !       statement would without leading and trailing spaces
;  197     0566  1 !       and gives that value to the destination string.
;  198     0567  1 !
;  199     0568  1 ! FORMAL PARAMETERS:
;  200     0569  1 !
;  201     0570  1 !       STRING.wt.dx            pointer to input string descriptor
;  202     0571  1 !       VALUE.rf.v              value of a floating number
;  203     0572  1 !
;  204     0573  1 ! IMPLICIT INPUTS:
;  205     0574  1 !
;  206     0575  1 !       NONE
;  207     0576  1 !
;  208     0577  1 ! IMPLICIT OUTPUTS:
;  209     0578  1 !
;  210     0579  1 !       NONE
;  211     0580  1 !
;  212     0581  1 ! ROUTINE VALUE:
;  213     0582  1 ! COMPLETION CODES:
;  214     0583  1 !
;  215     0584  1 !       NONE
;  216     0585  1 !
;  217     0586  1 ! SIDE EFFECTS:
;  218     0587  1 !
;  219     0588  1 !       This routine calls the conversion and so may signal any of its errors
;  220     0589  1 !       or have any of its side effects.  In particular, the conversion routine
;  221     0590  1 !       calls STR$ routines and so may allocate or deallocate dynamic string
;  222     0591  1 !       space, or write lock a string for a time.
;  223     0592  1 !
;  224     0593  1 !--
;  225     0594  1
;  226     0595  2     BEGIN
;  227     0596  2
;  228     0597  2     MAP
;  229     0598  2         STRING : REF BLOCK [8,BYTE];
;  230     0599  2
;  231     0600  2     LOCAL
;  232     0601  2         STR_LENGTH : WORD,                      ! conversion rtn returns len
;  233     0602  2         TEMP : VECTOR [2, LONG];                ! need double to pass to cnv
;  234     0603  2
;  235     0604  2     TEMP [0] = .VALUE;                          ! make value into double
;  236     0605  2     TEMP [1] = 0;
;  237     0606  2     BAS$CVT_OUT_D_G (TEMP [0],                  ! convert this value to string
;  238     0607  2                      strip_spaces,              ! set flag to strip spaces
;  239     0608  2                      STR_LENGTH,                ! return bytes needed for str
;  240     0609  2                      STRING [0,0,0,0]);         ! descriptor of result string
;  241     0610  2                                                 ! no scale to cvt
;  242     0611  2                                                 ! default # of digits
;  243     0612  2
```

```
;  244          0613  2     RETURN;
;  245          0614  1     END;                                    !End of BAS$STR_F


                                    0000 00000        .ENTRY  BAS$STR_F, Save nothing                 ; 0556
                               5E   0C   C2 00002     SUBL2   #12, SP                                 ;
                          04   AE   08   AC   D0 00005     MOVL    VALUE, TEMP                        ; 0604
                                    08   AE   D4 0000A     CLRL    TEMP+4                             ; 0605
                                    04   AC   DD 0000D     PUSHL   STRING                             ; 0609
                                    04   AE   9F 00010     PUSHAB  STR_LENGTH                         ; 0606
                                    01   DD 00013          PUSHL   #1                                 ; 0609
                                    10   AE   9F 00015     PUSHAB  TEMP                               ; 0606
                    00000000G   00           04   FB 00018     CALLS   #4, BAS$CVT_OUT_D_G            ; 0609
                                              04 0001F     RET                                        ; 0614

; Routine Size:  32 bytes,    Routine Base:  _BAS$CODE + 0021
```

```
;  247     0615  1 GLOBAL ROUTINE BAS$STR_D (                    ! convert double to string
;  248     0616  1                             STRING,            ! Address of destination descriptor
;  249     0617  1                             VALUE1,            ! 1st longword of double value to put in
;  250     0618  1                             VALUE2) :          ! 2nd longwrd of double value for string
;  251     0619  1                             NOVALUE =
;  252     0620  1
;  253     0621  1 !++
;  254     0622  1 ! FUNCTIONAL DESCRIPTION:
;  255     0623  1 !
;  256     0624  1 !       This routine takes a double number and formats it as the BASIC PRINT
;  257     0625  1 !       statement would, except without leading and trailing spaces,
;  258     0626  1 !       and gives that value to the destination string.
;  259     0627  1 !       Note that this routine violates the calling standard by accepting and
;  260     0628  1 !       calling a routine with double floating passed by value.
;  261     0629  1 !
;  262     0630  1 ! FORMAL PARAMETERS:
;  263     0631  1 !
;  264     0632  1 !       STRING.wt.dx            pointer to input string descriptor
;  265     0633  1 !       VALUE.rd.v              value of a double number
;  266     0634  1 !       (VALUE1 and VALUE2 used to pick up the 2 words of double value)
;  267     0635  1 !
;  268     0636  1 ! IMPLICIT INPUTS:
;  269     0637  1 !
;  270     0638  1 !       Scale factor from the BASIC frame
;  271     0639  1 !
;  272     0640  1 ! IMPLICIT OUTPUTS:
;  273     0641  1 !
;  274     0642  1 !       NONE
;  275     0643  1 !
;  276     0644  1 ! ROUTINE VALUE:
;  277     0645  1 ! COMPLETION CODES:
;  278     0646  1 !
;  279     0647  1 !       NONE
;  280     0648  1 !
;  281     0649  1 ! SIDE EFFECTS:
;  282     0650  1 !
;  283     0651  1 !       This routine calls the conversion routine and so may signal any of its
;  284     0652  1 !       errors and have any of its side effects.  In particular, the conversion
;  285     0653  1 !       routine calls STR$ routines and so may allocate or deallocate
;  286     0654  1 !       dynamic string space, or write lock a string for a short time.
;  287     0655  1 !
;  288     0656  1 !--
;  289     0657  1
;  290     0658  2     BEGIN
;  291     0659  2
;  292     0660  2     MAP
;  293     0661  2         STRING : REF BLOCK [8,BYTE];
;  294     0662  2
;  295     0663  2     LOCAL
;  296     0664  2         STR_LENGTH : WORD;                      ! conversion rtn returns len
;  297     0665  2
;  298     0666  2     BAS$CVT_OUT_D_G (VALUE1,                    ! convert this value to string
;  299     0667  2                      strip_spaces,              ! set flag to strip spaces
;  300     0668  2                      STR_LENGTH,                ! return bytes needed for str
;  301     0669  2                      STRING [0,0,0,0],          ! return string
;  302     0670  2                      $BAS$SCALE);               ! scale factor
;  303     0671  2                                                 ! default # of digits
```

```
;  304          0672  2
;  305          0673  2     RETURN;
;  306          0674  1     END;                          !End of BAS$STR_D


                                                         .EXTRN  BAS$$SCALE_L_R1

                                              0FFC 00000 .ENTRY  BAS$STR_D, Save R2,R3,R4,R5,R6,R7,R8,R9,-    ; 0615
                                                                 R10,R11                                       ;
                                        5E          04 C2 00002 SUBL2  #4, SP                                 ;
                                        51          5D D0 00005 MOVL   FP, FMP                                ; 0669
                                        50      0C  A1 D0 00008 MOVL   12(FMP), R0                            ;
                                    00000000G   00  16 0000C JSB    BAS$$SCALE_L_R1                           ;
                                                50  DD 00012 PUSHL  R0                                        ;
                                            04  AC  DD 00014 PUSHL  STRING                                    ;
                                            08  AE  9F 00017 PUSHAB STR_LENGTH                               ; 0666
                                                01  DD 0001A PUSHL  #1                                       ; 0669
                                            08  AC  9F 0001C PUSHAB VALUE1                                   ; 0666
                                00000000G   00      05  FB 0001F CALLS  #5, BAS$CVT_OUT_D_G                   ; 0669
                                                        04 00026 RET                                          ; 0674

; Routine Size:  39 bytes,    Routine Base:  _BAS$CODE + 0041
```

```
;  308     0675  1 GLOBAL ROUTINE BAS$STR_G (                    ! convert g float to string
;  309     0676  1                         STRING,                ! Address of destination descriptor
;  310     0677  1                         VALUE1,                ! 1st longword of g float value to put in
;  311     0678  1                         VALUE2) :              ! 2nd longwrd of g float value for string
;  312     0679  1                         NOVALUE =
;  313     0680  1
;  314     0681  1 !++
;  315     0682  1 ! FUNCTIONAL DESCRIPTION:
;  316     0683  1 !
;  317     0684  1 !       This routine takes a g float number and formats it as the BASIC PRINT
;  318     0685  1 !       statement would, except without leading and trailing spaces,
;  319     0686  1 !       and gives that value to the destination string.
;  320     0687  1 !       Note that this routine violates the calling standard by accepting and
;  321     0688  1 !       calling a routine with g floating passed by value.
;  322     0689  1 !
;  323     0690  1 ! FORMAL PARAMETERS:
;  324     0691  1 !
;  325     0692  1 !       STRING.wt.dx            pointer to input string descriptor
;  326     0693  1 !       VALUE.rg.v              value of a g float number
;  327     0694  1 !       (VALUE1 and VALUE2 used to pick up the 2 words of g float value)
;  328     0695  1 !
;  329     0696  1 ! IMPLICIT INPUTS:
;  330     0697  1 !
;  331     0698  1 !       NONE
;  332     0699  1 !
;  333     0700  1 ! IMPLICIT OUTPUTS:
;  334     0701  1 !
;  335     0702  1 !       NONE
;  336     0703  1 !
;  337     0704  1 ! ROUTINE VALUE:
;  338     0705  1 ! COMPLETION CODES:
;  339     0706  1 !
;  340     0707  1 !       NONE
;  341     0708  1 !
;  342     0709  1 ! SIDE EFFECTS:
;  343     0710  1 !
;  344     0711  1 !       This routine calls the conversion routine and so may signal any of its
;  345     0712  1 !       errors and have any of its side effects.  In particular, the conversion
;  346     0713  1 !       routine calls STR$ routines and so may allocate or deallocate
;  347     0714  1 !       dynamic string space, or write lock a string for a short time.
;  348     0715  1 !
;  349     0716  1 !--
;  350     0717  1
;  351     0718  2     BEGIN
;  352     0719  2
;  353     0720  2     MAP
;  354     0721  2         STRING : REF BLOCK [8,BYTE];
;  355     0722  2
;  356     0723  2     LOCAL
;  357     0724  2         STR_LENGTH : WORD;                      ! conversion rtn returns len
;  358     0725  2
;  359     0726  2     BAS$CVT_OUT_G_G (VALUE1,                    ! convert this value to string
;  360     0727  2                      strip_spaces,              ! set flag to strip spaces
;  361     0728  2                      STR_LENGTH,                ! return bytes needed for str
;  362     0729  2                      STRING [0,0,0,0]);         ! return string
;  363     0730  2                                                 ! default # of digits
;  364     0731  2
```

```
;  365          0732  2     RETURN;
;  366          0733  1     END;                            !End of BAS$STR_G



                                    0000 00000          .ENTRY  BAS$STR_G, Save nothing                 ; 0675
                                5E      04   C2 00002      SUBL2   #4, SP                                  ;
                             04     AC   DD 00005          PUSHL   STRING                                  ; 0729
                             04     AE   9F 00008          PUSHAB  STR_LENGTH                              ; 0726
                                    01   DD 0000B          PUSHL   #1                                      ; 0729
                             08     AC   9F 0000D          PUSHAB  VALUE1                                  ; 0726
                    00000000G   00      04   FB 00010      CALLS   #4, BAS$CVT_OUT_G_G                     ; 0729
                                        04 00017          RET                                             ; 0733

; Routine Size:  24 bytes,    Routine Base:  _BAS$CODE + 0068
```

```
;  368     0734  1 GLOBAL ROUTINE BAS$STR_H (                    ! convert g float to string
;  369     0735  1                             STRING,            ! Address of destination descriptor
;  370     0736  1                             VALUE1,            ! 1st longword of h float value to put in
;  371     0737  1                             VALUE2,            ! 2nd longword of h float value for string
;  372     0738  1                             VALUE3,            ! 3rd longword of h float value
;  373     0739  1                             VALUE4) :          ! 4th longword of h float value
;  374     0740  1                             NOVALUE =
;  375     0741  1
;  376     0742  1 !++
;  377     0743  1 ! FUNCTIONAL DESCRIPTION:
;  378     0744  1 !
;  379     0745  1 !       This routine takes an h float number and formats it as the BASIC PRINT
;  380     0746  1 !       statement would, except without leading and trailing spaces,
;  381     0747  1 !       and gives that value to the destination string.
;  382     0748  1 !       Note that this routine violates the calling standard by accepting and
;  383     0749  1 !       calling a routine with double floating passed by value.
;  384     0750  1 !
;  385     0751  1 ! FORMAL PARAMETERS:
;  386     0752  1 !
;  387     0753  1 !       STRING.wt.dx            pointer to input string descriptor
;  388     0754  1 !       VALUE.rg.v              value of a double number
;  389     0755  1 !       (VALUE1, VALUE2, VALUE3, & VALUE4 used to pick up the 4 words of h float value)
;  390     0756  1 !
;  391     0757  1 ! IMPLICIT INPUTS:
;  392     0758  1 !
;  393     0759  1 !       NONE
;  394     0760  1 !
;  395     0761  1 ! IMPLICIT OUTPUTS:
;  396     0762  1 !
;  397     0763  1 !       NONE
;  398     0764  1 !
;  399     0765  1 ! ROUTINE VALUE:
;  400     0766  1 ! COMPLETION CODES:
;  401     0767  1 !
;  402     0768  1 !       NONE
;  403     0769  1 !
;  404     0770  1 ! SIDE EFFECTS:
;  405     0771  1 !
;  406     0772  1 !       This routine calls the conversion routine and so may signal any of its
;  407     0773  1 !       errors and have any of its side effects.  In particular, the conversion
;  408     0774  1 !       routine calls STR$ routines and so may allocate or deallocate
;  409     0775  1 !       dynamic string space, or write lock a string for a short time.
;  410     0776  1 !
;  411     0777  1 !--
;  412     0778  1
;  413     0779  2     BEGIN
;  414     0780  2
;  415     0781  2     MAP
;  416     0782  2         STRING : REF BLOCK [8,BYTE];
;  417     0783  2
;  418     0784  2     LOCAL
;  419     0785  2         STR_LENGTH : WORD;                      ! conversion rtn returns len
;  420     0786  2
;  421     0787  2     BAS$CVT_OUT_H_G (VALUE1,                    ! convert this value to string
;  422     0788  2                      strip_spaces,              ! set flag to strip spaces
;  423     0789  2                      STR_LENGTH,                ! return bytes needed for str
;  424     0790  2                      STRING [0,0,0,0]);         ! return string
```

```
;  425          0791  2                                    ! default # of digits
;  426          0792  2
;  427          0793  2      RETURN;
;  428          0794  1      END;                    !End of BAS$STR_H


                                      0000 00000        .ENTRY  BAS$STR_H, Save nothing                 ; 0734
                              5E        04  C2 00002    SUBL2   #4, SP                                  ;
                          04  AC        DD 00005        PUSHL   STRING                                  ; 0790
                          04  AE        9F 00008        PUSHAB  STR_LENGTH                              ; 0787
                              01        DD 0000B        PUSHL   #1                                      ; 0790
                          08  AC        9F 0000D        PUSHAB  VALUE1                                  ; 0787
                  00000000G   00        04  FB 00010   CALLS   #4, BAS$CVT_OUT_H_G                     ; 0790
                                        04 00017        RET                                             ; 0794

; Routine Size:  24 bytes,    Routine Base:  _BAS$CODE + 0080
```

```
;  430     0795  1 GLOBAL ROUTINE BAS$STR_P (                    ! packed number to a string
;  431     0796  1                             STRING,            ! Address of destination descriptor
;  432     0797  1                             VALUE) :           ! Find numeric value of this string
;  433     0798  1                             NOVALUE =
;  434     0799  1
;  435     0800  1 !++
;  436     0801  1 ! FUNCTIONAL DESCRIPTION:
;  437     0802  1 !
;  438     0803  1 !       This routine takes a packed decimal number and formats it as the BASIC
;  439     0804  1 !       PRINT statement would without leading and trailing spaces
;  440     0805  1 !       and gives that value to the destination string.
;  441     0806  1 !
;  442     0807  1 ! FORMAL PARAMETERS:
;  443     0808  1 !
;  444     0809  1 !       STRING.wt.dx            pointer to input string descriptor
;  445     0810  1 !       VALUE.rp.dsd            desc of packed decimal number
;  446     0811  1 !
;  447     0812  1 ! IMPLICIT INPUTS:
;  448     0813  1 !
;  449     0814  1 !       NONE
;  450     0815  1 !
;  451     0816  1 ! IMPLICIT OUTPUTS:
;  452     0817  1 !
;  453     0818  1 !       NONE
;  454     0819  1 !
;  455     0820  1 ! ROUTINE VALUE:
;  456     0821  1 ! COMPLETION CODES:
;  457     0822  1 !
;  458     0823  1 !       NONE
;  459     0824  1 !
;  460     0825  1 ! SIDE EFFECTS:
;  461     0826  1 !
;  462     0827  1 !       This routine calls a conversion routine and so may signal any of its errors
;  463     0828  1 !       or have any of its side effects.  In particular, the conversion routine
;  464     0829  1 !       calls STR$ routines and so may allocate or deallocate dynamic string
;  465     0830  1 !       space, or write lock a string for a time.
;  466     0831  1 !
;  467     0832  1 !--
;  468     0833  1
;  469     0834  2     BEGIN
;  470     0835  2
;  471     0836  2     MAP
;  472     0837  2         STRING : REF BLOCK [8,BYTE],
;  473     0838  2         VALUE : REF BLOCK [12,BYTE];
;  474     0839  2
;  475     0840  2     LOCAL
;  476     0841  2         STR_LENGTH : WORD;                      ! conversion rtn returns len
;  477     0842  2
;  478     0843  2     BAS$CVT_OUT_P_G (.VALUE,                    ! convert this value to string
;  479     0844  2                      strip_spaces,              ! set flag to strip spaces
;  480     0845  2                      STR_LENGTH,                ! return bytes needed for str
;  481     0846  2                      STRING [0,0,0,0]);         ! descriptor of result string
;  482     0847  2                                                 ! no scale to cvt
;  483     0848  2                                                 ! default # of digits
;  484     0849  2
;  485     0850  2     RETURN;
;  486     0851  1     END;                                        !End of BAS$STR_P
```

```
                                 0000 00000     .ENTRY  BAS$STR_P, Save nothing                  ; 0795
                          5E       04   C2 00002     SUBL2   #4, SP                           ;
                       04  AC       DD 00005     PUSHL   STRING                           ; 0846
                       04  AE       9F 00008     PUSHAB  STR_LENGTH                       ; 0843
                           01       DD 0000B     PUSHL   #1                               ; 0846
                       08  AC       DD 0000D     PUSHL   VALUE                            ;
             00000000G     00       04   FB 00010     CALLS   #4, BAS$CVT_OUT_P_G              ;
                                     04 00017     RET                                      ; 0851

; Routine Size:  24 bytes,    Routine Base:  _BAS$CODE + 0098
```

```
;  488          0852  1 END                                !End of module
;  489          0853  0 ELUDOM



;                                  PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  _BAS$CODE                        176  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)



;                                  COMMAND QUALIFIERS

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

; Size:         176 code + 0 data bytes
; Run Time:        00:06.6
; Elapsed Time:    00:15.4
; Lines/CPU Min:    7766
; Lexemes/CPU-Min: 20922
; Memory Used:  38 pages
; Compilation Complete
```

BASUDFWF
LIS

BASSTRING
LIS

BASTERMIO
LIS

BASTRM
LIS

BASTAB
LIS

BASUDFRM
LIS

BASSYS
LIS

BASSTR
LIS

BASUDFRL
LIS