```
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
```

```
BBBBBBBB      AAAAAA      SSSSSSSS  NN      NN  UU      UU  MM      MM
BBBBBBBB      AAAAAA      SSSSSSSS  NN      NN  UU      UU  MM      MM
BB      BB   AA    AA   SS        NN      NN  UU      UU  MMMM  MMMM
BB      BB   AA    AA   SS        NN      NN  UU      UU  MMMM  MMMM
BB      BB   AA    AA   SS        NNNN    NN  UU      UU  MM  MM  MM
BB      BB   AA    AA   SS        NNNN    NN  UU      UU  MM  MM  MM
BBBBBBBB     AA    AA     SSSSSS   NN  NN  NN  UU      UU  MM      MM
BBBBBBBB     AA    AA     SSSSSS   NN  NN  NN  UU      UU  MM      MM
BB      BB   AAAAAAAAAA         SS  NN    NNNN  UU      UU  MM      MM
BB      BB   AAAAAAAAAA         SS  NN    NNNN  UU      UU  MM      MM
BB      BB   AA    AA           SS  NN      NN  UU      UU  MM      MM
BB      BB   AA    AA           SS  NN      NN  UU      UU  MM      MM
BBBBBBBB     AA    AA   SSSSSSSS    NN      NN  UUUUUUUUUU  MM      MM
BBBBBBBB     AA    AA   SSSSSSSS    NN      NN  UUUUUUUUUU  MM      MM

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$NUM (                      ! Routines to do BASIC NUM$ function
;     2          0002  0                 IDENT = '1-008'       ! module BASNUM.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 for long, floating, double,
;    37          0037  1 !       g floating ,h floating and packed decimal.
;    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 by a CALL to the BAS$ conversion routine.
;    42          0042  1 !
;    43          0043  1 ! ENVIRONMENT: User mode, AST level or not or mixed
;    44          0044  1 !
;    45          0045  1 ! AUTHOR: R. Will, CREATION DATE: 2-Mar-79
;    46          0046  1 !
;    47          0047  1 ! MODIFIED BY:
;    48          0048  1 !
;    49          0049  1 ! R. Will, 2-Mar-79: VERSION 01
;    50          0050  1 ! 01     - original
;    51          0051  1 ! 1-002 - Change string linkages to start with STR$.  JBS 04-JUN-1979
;    52          0052  1 ! 1-003 - Add BASLNK to get scaling linkages.  RW  26-JUN-79
;    53          0053  1 ! 1-004 - Change to use new conversion routines.  RW 6-JUL-79
;    54          0054  1 ! 1-005 - Add longword entry point.  RW  10-Sept-79
;    55          0055  1 ! 1-006 - String cleanup, don't use $STR$ macros.  R2  30-OCT-79
;    56          0056  1 ! 1-007 - Add entry points for G & H floating. PLL 3-Sep-81
;    57          0057  1 ! 1-008 - Add entry point for packed decimal.  PLL 19-Jan-82
```

```
;   58          0058  1 !--
;   59          0059  1 !<BLF/PAGE>
```

```
;    61          0060  1 !
;    62          0061  1 ! SWITCHES:
;    63          0062  1 !
;    64          0063  1
;    65          0064  1 SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
;    66          0065  1
;    67          0066  1 !
;    68          0067  1 ! LINKAGES: NONE
;    69          0068  1 !
;    70          0069  1
;    71          0070  1 !
;    72          0071  1 ! TABLE OF CONTENTS:
;    73          0072  1 !
;    74          0073  1
;    75          0074  1 FORWARD ROUTINE
;    76          0075  1     BAS$NUM_L : NOVALUE,                ! Find NUM$ of a longword value
;    77          0076  1     BAS$NUM_F : NOVALUE,                ! Find NUM$ of a floating value
;    78          0077  1     BAS$NUM_D : NOVALUE,                ! Find NUM$ of a double value
;    79          0078  1     BAS$NUM_G : NOVALUE,                ! Find NUM$ of a g floating value
;    80          0079  1     BAS$NUM_H : NOVALUE,                ! Find NUM$ of an h floating value
;    81          0080  1     BAS$NUM_P : NOVALUE;                ! Find NUM$ of a decimal value
;    82          0081  1
;    83          0082  1 !
;    84          0083  1 ! INCLUDE FILES:
;    85          0084  1 !
;    86          0085  1
;    87          0086  1 REQUIRE 'RTLIN:RTLPSECT';               ! Declare PSECTs code
;    88          0181  1 REQUIRE 'RTLIN:BASLNK';                 ! Linkage for BASIC scaling routines
;    89          0258  1 REQUIRE 'RTLIN:BASFRAME';               ! Define offsets in a BASIC frame
;    90          0461  1
;    91          0462  1 !
;    92          0463  1 ! MACROS: NONE
;    93          0464  1 !
;    94          0465  1
;    95          0466  1 !
;    96          0467  1 ! EQUATED SYMBOLS:
;    97          0468  1 !
;    98          0469  1
;    99          0470  1 LITERAL
;   100          0471  1     digits_in_long = 10,                ! # of significant digits to return
;   101          0472  1     no_flags = 0;                       ! no flags to conversion rtn
;   102          0473  1
;   103          0474  1 !
;   104          0475  1 ! PSECT DECLARATIONS
;   105          0476  1 !
;   106          0477  1
;   107          0478  1 DECLARE_PSECTS (BAS);
;   108          0479  1
;   109          0480  1 !
;   110          0481  1 ! OWN STORAGE: NONE
;   111          0482  1 !
;   112          0483  1
;   113          0484  1 !
;   114          0485  1 ! EXTERNAL REFERENCES:
;   115          0486  1 !
;   116          0487  1
;   117          0488  1 EXTERNAL ROUTINE
```

```
;  118          0489  1      BAS$CVT_OUT_D_G,                ! Convert dbl to BASIC string format
;  119          0490  1      BAS$CVT_OUT_G_G,                ! Convert gfloat to BASIC string format
;  120          0491  1      BAS$CVT_OUT_H_G,                ! Convert hfloat to BASIC string format
;  121          0492  1      BAS$CVT_OUT_P_G;                ! Convert packed to BASIC string format
;  122          0493  1
;  123          0494  1  BUILTIN
;  124          0495  1      CVTLD;                          ! convert long to double for cvt routine
```

```
;  126     0496  1 GLOBAL ROUTINE BAS$NUM_L (                    ! convert a longword to string
;  127     0497  1                            STRING,            ! Address of destination descriptor
;  128     0498  1                            VALUE) :           ! Create string with this value
;  129     0499  1                            NOVALUE =
;  130     0500  1
;  131     0501  1 !++
;  132     0502  1 ! FUNCTIONAL DESCRIPTION:
;  133     0503  1 !
;  134     0504  1 !       This routine takes a longword integer and formats it as the BASIC PRINT
;  135     0505  1 !       statement would and gives that value to the destination string.
;  136     0506  1 !
;  137     0507  1 ! FORMAL PARAMETERS:
;  138     0508  1 !
;  139     0509  1 !       STRING.wt.dx            pointer to input string descriptor
;  140     0510  1 !       VALUE.rl.v              value of a longword integer
;  141     0511  1 !
;  142     0512  1 ! IMPLICIT INPUTS:
;  143     0513  1 !
;  144     0514  1 !       NONE
;  145     0515  1 !
;  146     0516  1 ! IMPLICIT OUTPUTS:
;  147     0517  1 !
;  148     0518  1 !       NONE
;  149     0519  1 !
;  150     0520  1 ! ROUTINE VALUE:
;  151     0521  1 ! COMPLETION CODES:
;  152     0522  1 !
;  153     0523  1 !       NONE
;  154     0524  1 !
;  155     0525  1 ! SIDE EFFECTS:
;  156     0526  1 !
;  157     0527  1 !       Calls the conversion routine so may signal any of its errors or have
;  158     0528  1 !       any of its side effects.  In particular, the conversion routine calls
;  159     0529  1 !       some of the STR$ routines and so may allocate or deallocate space,
;  160     0530  1 !       and may keep some of the strings from being written for some period.
;  161     0531  1 !
;  162     0532  1 !--
;  163     0533  1
;  164     0534  2     BEGIN
;  165     0535  2
;  166     0536  2     MAP
;  167     0537  2         STRING : REF BLOCK [8,BYTE];
;  168     0538  2
;  169     0539  2     LOCAL
;  170     0540  2         STR_LENGTH : WORD,                      ! cvt returns str length
;  171     0541  2         TEMP : VECTOR [2, LONG];                ! need double for conversion
;  172     0542  2
;  173     0543  2     CVTLD (VALUE, TEMP [0]);                    ! convert to double
;  174     0544  2     BAS$CVT_OUT_D_G (TEMP [0],                  ! convert this value to string
;  175     0545  2                      no_flags,                  ! no flags needed
;  176     0546  2                      STR_LENGTH,                ! return bytes needed for str
;  177     0547  2                      STRING [0,0,0,0],          ! return string
;  178     0548  2                      0,                         ! no scale
;  179     0549  2                      digits_in_long);           ! digits to return
;  180     0550  2
;  181     0551  2     RETURN;
;  182     0552  1     END;                        !End of BAS$NUM_L
```

```
                                                        .TITLE  BAS$NUM
                                                        .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$NUM_L, Save nothing                 ; 0496
                                 5E       0C  C2 00002  SUBL2   #12, SP                                 ;
                              04 AE    08    AC  6E 00005  CVTLD   VALUE, TEMP                           ; 0543
                                             0A  DD 0000A  PUSHL   #10                                  ; 0547
                                             7E  D4 0000C  CLRL    -(SP)                                ;
                                          04 AC  DD 0000E  PUSHL   STRING                               ;
                                          0C AE  9F 00011  PUSHAB  STR_LENGTH                           ; 0544
                                             7E  D4 00014  CLRL    -(SP)                                ; 0547
                                          18 AE  9F 00016  PUSHAB  TEMP                                 ; 0544
                        00000000G     00       06  FB 00019  CALLS   #6, BAS$CVT_OUT_D_G               ; 0547
                                                04 00020  RET                                         ; 0552

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

```
;  184          0553  1 GLOBAL ROUTINE BAS$NUM_F (                    ! convert floating to string
;  185          0554  1                             STRING,           ! Address of destination descriptor
;  186          0555  1                             VALUE) :          ! Create string with this value
;  187          0556  1                             NOVALUE =
;  188          0557  1
;  189          0558  1 !++
;  190          0559  1 ! FUNCTIONAL DESCRIPTION:
;  191          0560  1 !
;  192          0561  1 !       This routine takes a floating number and formats it as the BASIC PRINT
;  193          0562  1 !       statement would and gives that value to the destination string.
;  194          0563  1 !
;  195          0564  1 ! FORMAL PARAMETERS:
;  196          0565  1 !
;  197          0566  1 !       STRING.wt.dx            pointer to input string descriptor
;  198          0567  1 !       VALUE.rf.v              value of a floating number
;  199          0568  1 !
;  200          0569  1 ! IMPLICIT INPUTS:
;  201          0570  1 !
;  202          0571  1 !       NONE
;  203          0572  1 !
;  204          0573  1 ! IMPLICIT OUTPUTS:
;  205          0574  1 !
;  206          0575  1 !       NONE
;  207          0576  1 !
;  208          0577  1 ! ROUTINE VALUE:
;  209          0578  1 ! COMPLETION CODES:
;  210          0579  1 !
;  211          0580  1 !       NONE
;  212          0581  1 !
;  213          0582  1 ! SIDE EFFECTS:
;  214          0583  1 !
;  215          0584  1 !       This routine calls the conversion routine and so may signal any of its
;  216          0585  1 !       errors or have any of its side effects.  In particular, the conversion
;  217          0586  1 !       routine calls STR$ routines and therefore may allocate or deallocate
;  218          0587  1 !       string space or lock a string from being written for some period.
;  219          0588  1 !
;  220          0589  1 !--
;  221          0590  1
;  222          0591  2     BEGIN
;  223          0592  2
;  224          0593  2     MAP
;  225          0594  2         STRING : REF BLOCK [8,BYTE];
;  226          0595  2
;  227          0596  2     LOCAL
;  228          0597  2         STR_LENGTH : WORD,                          ! cvt returns str length
;  229          0598  2         TEMP : VECTOR [2, LONG];                    ! need double for conversion
;  230          0599  2
;  231          0600  2     TEMP [0] = .VALUE;
;  232          0601  2     TEMP [1] = 0;
;  233          0602  2     BAS$CVT_OUT_D_G (TEMP [0],                      ! convert this value to string
;  234          0603  2                      no_flags,                      ! no flags needed
;  235          0604  2                      STR_LENGTH,                    ! return bytes needed for str
;  236          0605  2                      STRING [0,0,0,0]);             ! return string
;  237          0606  2                                                     ! no scale
;  238          0607  2                                                     ! default # of digits
;  239          0608  2
;  240          0609  2     RETURN;
```

```
;  241          0610  1     END;                                    !End of BAS$NUM_F



                                    0000 00000          .ENTRY  BAS$NUM_F, Save nothing                 ; 0553
                               5E   0C   C2 00002       SUBL2   #12, SP                                 ;
                           04  AE   08   AC   D0 00005  MOVL    VALUE, TEMP                             ; 0600
                               AE   08   AE   D4 0000A  CLRL    TEMP+4                                  ; 0601
                               AC   04   AC   DD 0000D  PUSHL   STRING                                  ; 0605
                               9F   04   AE   9F 00010  PUSHAB  STR_LENGTH                              ; 0602
                               D4   7E   D4 00013       CLRL    -(SP)                                   ; 0605
                               9F   10   AE   9F 00015  PUSHAB  TEMP                                    ; 0602
                    00000000G  00        04   FB 00018  CALLS   #4, BAS$CVT_OUT_D_G                     ; 0605
                                         04   04 0001F  RET                                            ; 0610

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

```
;  243     0611  1 GLOBAL ROUTINE BAS$NUM_D (                    ! convert double to string
;  244     0612  1                             STRING,            ! Address of destination descriptor
;  245     0613  1                             VALUE1,            ! 1st longword of double value to put in
;  246     0614  1                             VALUE2) :          ! 2nd longwrd of double value for string
;  247     0615  1                             NOVALUE =
;  248     0616  1
;  249     0617  1 !++
;  250     0618  1 ! FUNCTIONAL DESCRIPTION:
;  251     0619  1 !
;  252     0620  1 !       This routine takes a double number and formats it as the BASIC PRINT
;  253     0621  1 !       statement would and gives that value to the destination string.
;  254     0622  1 !       Note that this routine violates the calling standard because it
;  255     0623  1 !       accepts and passes double floating numbers by value.
;  256     0624  1 !
;  257     0625  1 ! FORMAL PARAMETERS:
;  258     0626  1 !
;  259     0627  1 !       STRING.wt.dx            pointer to input string descriptor
;  260     0628  1 !       VALUE.rd.v              value of a double number
;  261     0629  1 !       (VALUE1 and VALUE2 used to pick up the 2 longwords of double value)
;  262     0630  1 !
;  263     0631  1 ! IMPLICIT INPUTS:
;  264     0632  1 !
;  265     0633  1 !       Scale factor from BASIC frame
;  266     0634  1 !
;  267     0635  1 ! IMPLICIT OUTPUTS:
;  268     0636  1 !
;  269     0637  1 !       NONE
;  270     0638  1 !
;  271     0639  1 ! ROUTINE VALUE:
;  272     0640  1 ! COMPLETION CODES:
;  273     0641  1 !
;  274     0642  1 !       NONE
;  275     0643  1 !
;  276     0644  1 ! SIDE EFFECTS:
;  277     0645  1 !
;  278     0646  1 !       This routine calls the conversion routine and so may signal any of
;  279     0647  1 !       its errors or have any of its side effects.  In particular, the
;  280     0648  1 !       conversion routine calls the STR$ routines and so may allocate or
;  281     0649  1 !       deallocate dynamic string space, and may lock a string against
;  282     0650  1 !       writing for a period of time.
;  283     0651  1 !
;  284     0652  1 !--
;  285     0653  1
;  286     0654  2     BEGIN
;  287     0655  2
;  288     0656  2     MAP
;  289     0657  2         STRING : REF BLOCK [8,BYTE];
;  290     0658  2
;  291     0659  2     LOCAL
;  292     0660  2         STR_LENGTH : WORD;                      ! conversion rtn returns len
;  293     0661  2
;  294     0662  2     BAS$CVT_OUT_D_G (VALUE1,                    ! convert this value to string
;  295     0663  2                      no_flags,                  ! no flags needed
;  296     0664  2                      STR_LENGTH,                ! return bytes needed for str
;  297     0665  2                      STRING [0,0,0,0],          ! return string
;  298     0666  2                      $BAS$SCALE);               ! scale factor
;  299     0667  2                                                 ! default # of digits
```

```
;  300          0668  2
;  301          0669  2     RETURN;
;  302          0670  1     END;                        !End of BAS$NUM_D


                                                        .EXTRN  BAS$$SCALE_L_R1

                                        0FFC 00000      .ENTRY  BAS$NUM_D, Save R2,R3,R4,R5,R6,R7,R8,R9,-    ; 0611
                                                                R10,R11                                      ;
                                    5E      04  C2 00002        SUBL2   #4, SP                               ;
                                    51      5D  D0 00005        MOVL    FP, FMP                              ; 0665
                                    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                         ; 0662
                                                    7E D4 0001A CLRL    -(SP)                                ; 0665
                                                08  AC 9F 0001C PUSHAB  VALUE1                              ; 0662
                                00000000G   00      05  FB 0001F CALLS   #5, BAS$CVT_OUT_D_G                ; 0665
                                                        04 00026 RET                                         ; 0670

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

```
;  304     0671  1 GLOBAL ROUTINE BAS$NUM_G (                    ! convert g floating to string
;  305     0672  1                             STRING,            ! Address of destination descriptor
;  306     0673  1                             VALUE1,            ! 1st longword of g floating
;  307     0674  1                             VALUE2) :          ! 2nd longword of g floating
;  308     0675  1                             NOVALUE =
;  309     0676  1
;  310     0677  1 !++
;  311     0678  1 ! FUNCTIONAL DESCRIPTION:
;  312     0679  1 !
;  313     0680  1 !       This routine takes a g floating number and formats it as the BASIC PRINT
;  314     0681  1 !       statement would and gives that value to the destination string.
;  315     0682  1 !
;  316     0683  1 ! FORMAL PARAMETERS:
;  317     0684  1 !
;  318     0685  1 !       STRING.wt.dx            pointer to input string descriptor
;  319     0686  1 !       VALUE.rg.v              value of a g floating number
;  320     0687  1 !       (VALUE1 and VALUE2 used to pick up the 2 longwords of g floating)
;  321     0688  1 !
;  322     0689  1 ! IMPLICIT INPUTS:
;  323     0690  1 !
;  324     0691  1 !       NONE
;  325     0692  1 !
;  326     0693  1 ! IMPLICIT OUTPUTS:
;  327     0694  1 !
;  328     0695  1 !       NONE
;  329     0696  1 !
;  330     0697  1 ! ROUTINE VALUE:
;  331     0698  1 ! COMPLETION CODES:
;  332     0699  1 !
;  333     0700  1 !       NONE
;  334     0701  1 !
;  335     0702  1 ! SIDE EFFECTS:
;  336     0703  1 !
;  337     0704  1 !       This routine calls the conversion routine and so may signal any of its
;  338     0705  1 !       errors or have any of its side effects.  In particular, the conversion
;  339     0706  1 !       routine calls STR$ routines and therefore may allocate or deallocate
;  340     0707  1 !       string space or lock a string from being written for some period.
;  341     0708  1 !
;  342     0709  1 !--
;  343     0710  1
;  344     0711  2     BEGIN
;  345     0712  2
;  346     0713  2     MAP
;  347     0714  2         STRING : REF BLOCK [8,BYTE];
;  348     0715  2
;  349     0716  2     LOCAL
;  350     0717  2         STR_LENGTH : WORD;                      ! cvt returns str length
;  351     0718  2
;  352     0719  2     BAS$CVT_OUT_G_G (VALUE1,                    ! convert this value to string
;  353     0720  2                      no_flags,                  ! no flags needed
;  354     0721  2                      STR_LENGTH,                ! return bytes needed for str
;  355     0722  2                      STRING [0,0,0,0]);         ! return string
;  356     0723  2                                                 ! no scale
;  357     0724  2                                                 ! default # of digits
;  358     0725  2
;  359     0726  2     RETURN;
;  360     0727  1     END;                                        !End of BAS$NUM_G
```

```
                                 0000 00000     .ENTRY  BAS$NUM_G, Save nothing                  ; 0671
                          5E       04 00002     SUBL2   #4, SP                                  ;
                       04 AC       DD 00005     PUSHL   STRING                                  ; 0722
                       04 AE       9F 00008     PUSHAB  STR_LENGTH                              ; 0719
                          7E       D4 0000B     CLRL    -(SP)                                   ; 0722
                       08 AC       9F 0000D     PUSHAB  VALUE1                                  ; 0719
                00000000G  00      04 FB 00010  CALLS   #4, BAS$CVT_OUT_G_G                     ; 0722
                                      04 00017  RET                                             ; 0727

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

```
;  362     0728  1 GLOBAL ROUTINE BAS$NUM_H (                    ! convert h floating to string
;  363     0729  1                             STRING,          ! Address of destination descriptor
;  364     0730  1                             VALUE1,          ! 1st longword of h floating
;  365     0731  1                             VALUE2,          ! 2nd longword of h floating
;  366     0732  1                             VALUE3,          ! 3rd longword of h floating
;  367     0733  1                             VALUE4) :        ! 4th longword of h floating
;  368     0734  1                             NOVALUE =
;  369     0735  1
;  370     0736  1 !++
;  371     0737  1 ! FUNCTIONAL DESCRIPTION:
;  372     0738  1 !
;  373     0739  1 !       This routine takes an h floating number and formats it as the BASIC PRINT
;  374     0740  1 !       statement would and gives that value to the destination string.
;  375     0741  1 !
;  376     0742  1 ! FORMAL PARAMETERS:
;  377     0743  1 !
;  378     0744  1 !       STRING.wt.dx            pointer to input string descriptor
;  379     0745  1 !       VALUE.rh.v              value of an h floating number
;  380     0746  1 !       (Four longwords needed to pass an h floating number)
;  381     0747  1 !
;  382     0748  1 ! IMPLICIT INPUTS:
;  383     0749  1 !
;  384     0750  1 !       NONE
;  385     0751  1 !
;  386     0752  1 ! IMPLICIT OUTPUTS:
;  387     0753  1 !
;  388     0754  1 !       NONE
;  389     0755  1 !
;  390     0756  1 ! ROUTINE VALUE:
;  391     0757  1 ! COMPLETION CODES:
;  392     0758  1 !
;  393     0759  1 !       NONE
;  394     0760  1 !
;  395     0761  1 ! SIDE EFFECTS:
;  396     0762  1 !
;  397     0763  1 !       This routine calls the conversion routine and so may signal any of its
;  398     0764  1 !       errors or have any of its side effects.  In particular, the conversion
;  399     0765  1 !       routine calls STR$ routines and therefore may allocate or deallocate
;  400     0766  1 !       string space or lock a string from being written for some period.
;  401     0767  1 !
;  402     0768  1 !--
;  403     0769  1
;  404     0770  2     BEGIN
;  405     0771  2
;  406     0772  2     MAP
;  407     0773  2         STRING : REF BLOCK [8,BYTE];
;  408     0774  2
;  409     0775  2     LOCAL
;  410     0776  2         STR_LENGTH : WORD;                      ! cvt returns str length
;  411     0777  2
;  412     0778  2     BAS$CVT_OUT_H_G (VALUE1,                    ! convert this value to string
;  413     0779  2                      no_flags,                  ! no flags needed
;  414     0780  2                      STR_LENGTH,                ! return bytes needed for str
;  415     0781  2                      STRING [0,0,0,0]);         ! return string
;  416     0782  2                                                 ! no scale
;  417     0783  2                                                 ! default # of digits
;  418     0784  2
```

```
;  419          0785  2     RETURN;
;  420          0786  1     END;                                    !End of BAS$NUM_H



                                    0000 00000      .ENTRY  BAS$NUM_H, Save nothing                 ; 0728
                              5E    04   C2 00002    SUBL2   #4, SP                                  ;
                           04 AC    DD   00005      PUSHL   STRING                                  ; 0781
                           04 AE    9F   00008      PUSHAB  STR_LENGTH                              ; 0778
                                    7E   D4 0000B    CLRL    -(SP)                                   ; 0781
                           08 AC    9F   0000D      PUSHAB  VALUE1                                  ; 0778
                00000000G  00       04   FB 00010    CALLS   #4, BAS$CVT_OUT_H_G                     ; 0781
                                         04 00017    RET                                             ; 0786

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

```
;  422     0787  1 GLOBAL ROUTINE BAS$NUM_P (                    ! convert packed decimal to string
;  423     0788  1                             STRING,          ! Address of destination descriptor
;  424     0789  1                             VALUE) :         ! Create string with this value
;  425     0790  1                             NOVALUE =
;  426     0791  1
;  427     0792  1 !++
;  428     0793  1 ! FUNCTIONAL DESCRIPTION:
;  429     0794  1 !
;  430     0795  1 !       This routine takes a decimal number and formats it as the BASIC PRINT
;  431     0796  1 !       statement would and gives that value to the destination string.
;  432     0797  1 !
;  433     0798  1 ! FORMAL PARAMETERS:
;  434     0799  1 !
;  435     0800  1 !       STRING.wt.dx            pointer to input string descriptor
;  436     0801  1 !       VALUE.rp.dsd            desc of packed decimal number
;  437     0802  1 !
;  438     0803  1 ! IMPLICIT INPUTS:
;  439     0804  1 !
;  440     0805  1 !       NONE
;  441     0806  1 !
;  442     0807  1 ! IMPLICIT OUTPUTS:
;  443     0808  1 !
;  444     0809  1 !       NONE
;  445     0810  1 !
;  446     0811  1 ! ROUTINE VALUE:
;  447     0812  1 ! COMPLETION CODES:
;  448     0813  1 !
;  449     0814  1 !       NONE
;  450     0815  1 !
;  451     0816  1 ! SIDE EFFECTS:
;  452     0817  1 !
;  453     0818  1 !       This routine calls the conversion routine and so may signal any of its
;  454     0819  1 !       errors or have any of its side effects.  In particular, the conversion
;  455     0820  1 !       routine calls STR$ routines and therefore may allocate or deallocate
;  456     0821  1 !       string space or lock a string from being written for some period.
;  457     0822  1 !
;  458     0823  1 !--
;  459     0824  1
;  460     0825  2     BEGIN
;  461     0826  2
;  462     0827  2     MAP
;  463     0828  2         STRING : REF BLOCK [8,BYTE],
;  464     0829  2         VALUE : REF BLOCK [12,BYTE];
;  465     0830  2
;  466     0831  2     LOCAL
;  467     0832  2         STR_LENGTH : WORD;                      ! cvt returns str length
;  468     0833  2
;  469     0834  2     BAS$CVT_OUT_P_G (.VALUE,                    ! convert this value to string
;  470     0835  2                      no_flags,                  ! no flags needed
;  471     0836  2                      STR_LENGTH,                ! return bytes needed for str
;  472     0837  2                      STRING [0,0,0,0]);         ! return string
;  473     0838  2                                                 ! no scale
;  474     0839  2                                                 ! default # of digits
;  475     0840  2
;  476     0841  2     RETURN;
;  477     0842  1     END;                                        !End of BAS$NUM_P
```

```
                                    0000 00000          .ENTRY  BAS$NUM_P, Save nothing                 ; 0787
                              5E          04  C2 00002 SUBL2   #4, SP                                  ;
                           04     04  AC  DD 00005 PUSHL   STRING                                     ; 0837
                           04     04  AE  9F 00008 PUSHAB  STR_LENGTH                                 ; 0834
                                      7E  D4 0000B CLRL    -(SP)                                      ; 0837
                           08     08  AC  DD 0000D PUSHL   VALUE                                      ;
                 00000000G    00      04  FB 00010 CALLS   #4, BAS$CVT_OUT_P_G                       ;
                                          04 00017 RET                                               ; 0842

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

```
;  479          0843  1 END                                !End of module
;  480          0844  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$:BASNUM/OBJ=OBJ$:BASNUM MSRC$:BASNUM/UPDATE=(ENH$:BASNUM)

; Size:         176 code + 0 data bytes
; Run Time:         00:06.6
; Elapsed Time:     00:13.2
; Lines/CPU Min:     7696
; Lexemes/CPU-Min: 20954
; Memory Used:  38 pages
; Compilation Complete
```

BASMID
LIS

BASMULDDI
LIS

BASNOTIMP
LIS

BASMOVEAR
LIS

BASMSGDEF
LIS

BASMSGGEN
LIS

BASONECHR
LIS

BASMOVE
LIS

BASNUM
LIS

BASNAMEAS
LIS

BASNUM1
LIS