```
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      11
BBBBBBBB      AAAAAA      SSSSSSSS  NN      NN  UU      UU  MM      MM      11
BB      BB  AA      AA  SS        NN      NN  UU      UU  MMMM  MMMM    1111
BB      BB  AA      AA  SS        NN      NN  UU      UU  MMMM  MMMM    1111
BB      BB  AA      AA  SS        NNNN    NN  UU      UU  MM  MM  MM      11
BB      BB  AA      AA  SS        NNNN    NN  UU      UU  MM  MM  MM      11
BBBBBBBB    AA      AA    SSSSSS  NN  NN  NN  UU      UU  MM      MM      11
BBBBBBBB    AA      AA    SSSSSS  NN  NN  NN  UU      UU  MM      MM      11
BB      BB  AAAAAAAAAA          SS  NN    NNNN  UU      UU  MM      MM      11
BB      BB  AAAAAAAAAA          SS  NN    NNNN  UU      UU  MM      MM      11
BB      BB  AA      AA          SS  NN      NN  UU      UU  MM      MM      11      ....
BB      BB  AA      AA          SS  NN      NN  UU      UU  MM      MM      11      ....
BBBBBBBB    AA      AA  SSSSSSSS  NN      NN  UUUUUUUUUU  MM      MM    111111    ....
BBBBBBBB    AA      AA  SSSSSSSS  NN      NN  UUUUUUUUUU  MM      MM    111111    ....
```

```
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$NUM1 (                    ! Routines to do BASIC NUM1$ function
;     2          0002  0                 IDENT = '1-008'      ! module BASNUM1.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 both routines convert a number
;    40          0040  1 !       to a numeric string with no leading or trailing spaces and no E format
;    41          0041  1 !       by a CALL to the correct 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: 10-Mar-79
;    46          0046  1 !
;    47          0047  1 ! MODIFIED BY:
;    48          0048  1 !
;    49          0049  1 ! R. Will, 10-Mar-79: VERSION 01
;    50          0050  1 ! 01     - original
;    51          0051  1 ! 1-002 - Change linkage names for string linkages to STR$.  JBS 04-JUN-1979
;    52          0052  1 ! 1-003 - Add BASLNK for BASIC scaling linkages.  RW  26-JUN-79
;    53          0053  1 ! 1-004 - Change to use new conversion routines.  RW  6-JUL-79
;    54          0054  1 ! 1-005 - Add longword entry point.  RW  10-Jul-79
;    55          0055  1 ! 1-006 - String cleanup, don't use STR$ macros.  RW  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$NUM1_F : NOVALUE,                   ! Find NUM1$ of a floating value
;    77          0076  1     BAS$NUM1_D : NOVALUE,                   ! Find NUM1$ of a double value
;    78          0077  1     BAS$NUM1_L : NOVALUE,                   ! Find NUM1$ of a longword value
;    79          0078  1     BAS$NUM1_G : NOVALUE,                   ! Find NUM1$ of a g float value
;    80          0079  1     BAS$NUM1_H : NOVALUE,                   ! Find NUM1$ of an h float value
;    81          0080  1     BAS$NUM1_P : NOVALUE;                   ! Find NUM1$ 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 BUILTIN
;    96          0467  1     CVTLD;                                  ! convert longword to double precision
;    97          0468  1
;    98          0469  1 !
;    99          0470  1 ! EQUATED SYMBOLS:
;   100          0471  1 !
;   101          0472  1
;   102          0473  1 LITERAL
;   103          0474  1     strip_spaces = 1,                       ! flag to strip spaces
;   104          0475  1     floating_dec = 32;                      ! flag to use floating decimal point
;   105          0476  1
;   106          0477  1 !
;   107          0478  1 ! PSECT DECLARATIONS
;   108          0479  1 !
;   109          0480  1
;   110          0481  1 DECLARE_PSECTS (BAS);
;   111          0482  1
;   112          0483  1 !
;   113          0484  1 ! OWN STORAGE: NONE
;   114          0485  1 !
;   115          0486  1
;   116          0487  1 !
;   117          0488  1 ! EXTERNAL REFERENCES:
```

```
;  118          0489  1 !
;  119          0490  1
;  120          0491  1 EXTERNAL ROUTINE
;  121          0492  1     BAS$CVT_OUT_F_F,                ! Convert # to BASIC string format
;  122          0493  1     BAS$CVT_OUT_D_F,                ! Convert # to BASIC double str format
;  123          0494  1     BAS$CVT_OUT_G_F,                ! Convert # to BASIC g float str format
;  124          0495  1     BAS$CVT_OUT_H_F,                ! Convert # to BASIC h float str format
;  125          0496  1     BAS$CVT_OUT_P_F;                ! Convert # to BASIC decimal str format
```

```
;  127     0497  1 GLOBAL ROUTINE BAS$NUM1_F (                  ! convert floating to string
;  128     0498  1                             STRING,           ! Address of destination descriptor
;  129     0499  1                             VALUE) :          ! Create string with this value
;  130     0500  1                             NOVALUE =
;  131     0501  1
;  132     0502  1 !++
;  133     0503  1 ! FUNCTIONAL DESCRIPTION:
;  134     0504  1 !
;  135     0505  1 !       This routine takes a floating number and formats it
;  136     0506  1 !       with no leading or trailing spaces and no E format, 6 significant
;  137     0507  1 !       digits, and gives that value to the destination string.
;  138     0508  1 !
;  139     0509  1 ! FORMAL PARAMETERS:
;  140     0510  1 !
;  141     0511  1 !       STRING.wt.dx            pointer to input string descriptor
;  142     0512  1 !       VALUE.rf.v              value of a floating number
;  143     0513  1 !
;  144     0514  1 ! IMPLICIT INPUTS:
;  145     0515  1 !
;  146     0516  1 !       NONE
;  147     0517  1 !
;  148     0518  1 ! IMPLICIT OUTPUTS:
;  149     0519  1 !
;  150     0520  1 !       NONE
;  151     0521  1 !
;  152     0522  1 ! ROUTINE VALUE:
;  153     0523  1 ! COMPLETION CODES:
;  154     0524  1 !
;  155     0525  1 !       NONE
;  156     0526  1 !
;  157     0527  1 ! SIDE EFFECTS:
;  158     0528  1 !
;  159     0529  1 !       This routine calls the conversion routine and so may signal any of its
;  160     0530  1 !       errors or have any of its side effects.  In particular, the conversion
;  161     0531  1 !       routine calls STR$ routines and so may allocate or deallocate
;  162     0532  1 !       dynamic string space and write lock a string for a short time.
;  163     0533  1 !
;  164     0534  1 !--
;  165     0535  1
;  166     0536  2     BEGIN
;  167     0537  2
;  168     0538  2     MAP
;  169     0539  2         STRING : REF BLOCK [8,BYTE];
;  170     0540  2
;  171     0541  2     LOCAL
;  172     0542  2         STR_LENGTH : WORD;                      ! cvt returns str lengtht
;  173     0543  2
;  174     0544  2     BAS$CVT_OUT_F_F (VALUE,                     ! convert this value to string
;  175     0545  2                      0,                         ! don't specify integer digits
;  176     0546  2                      0,                         ! don't specify fraction digits
;  177     0547  2                      strip_spaces + floating_dec,! set flags
;  178     0548  2                      STR_LENGTH,                ! return bytes needed for str
;  179     0549  2                      STRING [0,0,0,0]);         ! return string
;  180     0550  2
;  181     0551  2     RETURN;
;  182     0552  1     END;                                        !End of BAS$NUM1_F
```

```
                                                        .TITLE  BAS$NUM1
                                                        .IDENT  \1-008\

                                                        .EXTRN  BAS$CVT_OUT_F_F
                                                        .EXTRN  BAS$CVT_OUT_D_F
                                                        .EXTRN  BAS$CVT_OUT_G_F
                                                        .EXTRN  BAS$CVT_OUT_H_F
                                                        .EXTRN  BAS$CVT_OUT_P_F

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

                                  0000 00000            .ENTRY  BAS$NUM1_F, Save nothing                ; 0497
                            5E          04  C2 00002   SUBL2   #4, SP                                  ;
                                    04  AC  DD 00005   PUSHL   STRING                                  ; 0549
                                    04  AE  9F 00008   PUSHAB  STR_LENGTH                              ; 0544
                                        21  DD 0000B   PUSHL   #33                                    ; 0549
                                        7E  7C 0000D   CLRQ    -(SP)                                   ;
                                    08  AC  9F 0000F   PUSHAB  VALUE                                   ; 0544
                00000000G   00          06  FB 00012   CALLS   #6, BAS$CVT_OUT_F_F                     ; 0549
                                            04 00019   RET                                             ; 0552

; Routine Size:  26 bytes,    Routine Base:  _BAS$CODE + 0000
```

```
;  184     0553  1 GLOBAL ROUTINE BAS$NUM1_D (                  ! convert double to string
;  185     0554  1                             STRING,           ! Address of destination descriptor
;  186     0555  1                             VALUE1,           ! 1st longword of double value to put in
;  187     0556  1                             VALUE2) :         ! 2nd longwrd of double value for string
;  188     0557  1                             NOVALUE =
;  189     0558  1
;  190     0559  1 !++
;  191     0560  1 ! FUNCTIONAL DESCRIPTION:
;  192     0561  1 !
;  193     0562  1 !       This routine takes a double number and formats it without
;  194     0563  1 !       leading or trailing spaces, and no E format, 16 significant digits,
;  195     0564  1 !       and gives that value to the destination string.
;  196     0565  1 !       Note that this routine violates the calling standard by accepting
;  197     0566  1 !       and passing double floating numbers by value
;  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.rd.v              value of a double number
;  203     0572  1 !       (VALUE1 and VALUE2 used to pick up the 2 longwords of double value)
;  204     0573  1 !
;  205     0574  1 ! IMPLICIT INPUTS:
;  206     0575  1 !
;  207     0576  1 !       scale from callers frame for double scaling
;  208     0577  1 !
;  209     0578  1 ! IMPLICIT OUTPUTS:
;  210     0579  1 !
;  211     0580  1 !       NONE
;  212     0581  1 !
;  213     0582  1 ! ROUTINE VALUE:
;  214     0583  1 ! COMPLETION CODES:
;  215     0584  1 !
;  216     0585  1 !       NONE
;  217     0586  1 !
;  218     0587  1 ! SIDE EFFECTS:
;  219     0588  1 !
;  220     0589  1 !       This routine calls the conversion routine and so may signal any of its
;  221     0590  1 !       errors or have any of its side effects.  In particular, the conversion
;  222     0591  1 !       routine calls STR$ routines and so may allocate or deallocate
;  223     0592  1 !       dynamic string space and write lock a string for a short time.
;  224     0593  1 !
;  225     0594  1 !--
;  226     0595  1
;  227     0596  2     BEGIN
;  228     0597  2
;  229     0598  2     MAP
;  230     0599  2         STRING : REF BLOCK [8,BYTE];
;  231     0600  2
;  232     0601  2     LOCAL
;  233     0602  2         STR_LENGTH : WORD;                          ! conversion rtn returns len
;  234     0603  2
;  235     0604  2     BAS$CVT_OUT_D_F (VALUE1,                        ! convert this value to string
;  236     0605  2                      0,                             ! don't specify integer digits
;  237     0606  2                      0,                             ! don't specify fraction digits
;  238     0607  2                      strip_spaces + floating_dec,   ! set flags
;  239     0608  2                      STR_LENGTH,                    ! return bytes needed for str
;  240     0609  2                      STRING [0,0,0,0],              ! return string
```

```
;  241          0610  2                    $BAS$SCALE);                    ! scale factor
;  242          0611  2
;  243          0612  2        RETURN;
;  244          0613  1        END;                                        !End of BAS$NUM1_D


                                                                           .EXTRN  BAS$$SCALE_L_R1

                                                                           .ENTRY  BAS$NUM1_D, Save R2,R3,R4,R5,R6,R7,R8,R9,-  ; 0553
                                                                                   R10,R11
                                                                           SUBL2   #4, SP                                      ;
                                                                           MOVL    FP, FMP                                     ; 0609
                                                                           MOVL    12(FMP), R0                                 ;
                                                                           JSB     BAS$$SCALE_L_R1                             ;
                                                                           PUSHL   R0                                          ;
                                                                           PUSHL   STRING                                      ;
                                                                           PUSHAB  STR_LENGTH                                  ; 0604
                                                                           PUSHL   #33                                         ; 0609
                                                                           CLRQ    -(SP)                                       ;
                                                                           PUSHAB  VALUE1                                      ; 0604
                                                                           CALLS   #7, BAS$CVT_OUT_D_F                         ; 0609
                                                                           RET                                                 ; 0613

; Routine Size:  41 bytes,    Routine Base:  _BAS$CODE + 001A
```

```
;  246     0614  1 GLOBAL ROUTINE BAS$NUM1_L (                  ! convert longword to string
;  247     0615  1                             STRING,           ! Address of destination descriptor
;  248     0616  1                             VALUE) :          ! longword value to put in string
;  249     0617  1                             NOVALUE =
;  250     0618  1
;  251     0619  1 !++
;  252     0620  1 ! FUNCTIONAL DESCRIPTION:
;  253     0621  1 !
;  254     0622  1 !       This routine takes a longword number and formats it without
;  255     0623  1 !       leading or trailing spaces, and no E format, 16 significant digits,
;  256     0624  1 !       and gives that value to the destination string.
;  257     0625  1 !
;  258     0626  1 ! FORMAL PARAMETERS:
;  259     0627  1 !
;  260     0628  1 !       STRING.wt.dx            pointer to input string descriptor
;  261     0629  1 !       VALUE.rl.v              value of a double number
;  262     0630  1 !
;  263     0631  1 ! IMPLICIT INPUTS:
;  264     0632  1 !
;  265     0633  1 !       NONE
;  266     0634  1 !
;  267     0635  1 ! IMPLICIT OUTPUTS:
;  268     0636  1 !
;  269     0637  1 !       NONE
;  270     0638  1 !
;  271     0639  1 ! ROUTINE VALUE:
;  272     0640  1 ! COMPLETION CODES:
;  273     0641  1 !
;  274     0642  1 !       NONE
;  275     0643  1 !
;  276     0644  1 ! SIDE EFFECTS:
;  277     0645  1 !
;  278     0646  1 !       This routine calls the conversion routine and so may signal any of its
;  279     0647  1 !       errors or have any of its side effects.  In particular, the conversion
;  280     0648  1 !       routine calls STR$ routines and so may allocate or deallocate
;  281     0649  1 !       dynamic string space and write lock a string for a short time.
;  282     0650  1 !
;  283     0651  1 !--
;  284     0652  1
;  285     0653  2     BEGIN
;  286     0654  2
;  287     0655  2     MAP
;  288     0656  2         STRING : REF BLOCK [8,BYTE];
;  289     0657  2
;  290     0658  2     LOCAL
;  291     0659  2         STR_LENGTH : WORD,                          ! conversion rtn returns len
;  292     0660  2         DOUBLE_VALUE : VECTOR [2, LONG];            ! place to store double
;  293     0661  2
;  294     0662  2     CVTLD (VALUE, DOUBLE_VALUE [0]);                ! convert logn to double and
;  295     0663  2                                                     !  then use double conversion
;  296     0664  2     BAS$CVT_OUT_D_F (DOUBLE_VALUE [0],              ! double value to convert
;  297     0665  2                      0,                             ! don't specify integer digits
;  298     0666  2                      0,                             ! don't specify fraction digits
;  299     0667  2                      strip_spaces + floating_dec,   ! set flags
;  300     0668  2                      STR_LENGTH,                    ! return bytes needed for str
;  301     0669  2                      STRING [0,0,0,0]);             ! return string
;  302     0670  2
```

```
;  303          0671  2     RETURN;
;  304          0672  1     END;                        !End of BAS$NUM1_L



                                    0000 00000          .ENTRY  BAS$NUM1_L, Save nothing                ; 0614
                              5E    0C   C2 00002        SUBL2   #12, SP                                 ;
                        04    AE    08   AC  6E 00005    CVTLD   VALUE, DOUBLE_VALUE                     ; 0662
                                    04   AC  DD 0000A    PUSHL   STRING                                  ; 0669
                                    04   AE  9F 0000D    PUSHAB  STR_LENGTH                              ; 0664
                                    21   DD  00010        PUSHL   #33                                     ; 0669
                                    7E   7C  00012        CLRQ    -(SP)                                   ;
                                    18   AE  9F 00014    PUSHAB  DOUBLE_VALUE                            ; 0664
                  00000000G   00            06   FB 00017    CALLS   #6, BAS$CVT_OUT_D_F                     ; 0669
                                                04 0001E    RET                                             ; 0672

; Routine Size:  31 bytes,    Routine Base:  _BAS$CODE + 0043
```

```
;  306     0673  1 GLOBAL ROUTINE BAS$NUM1_G (                  ! convert g float to string
;  307     0674  1                              STRING,           ! Address of destination descriptor
;  308     0675  1                              VALUE1,           ! 1st longword of g float value to put in
;  309     0676  1                              VALUE2) :         ! 2nd longwrd of g float value for string
;  310     0677  1                              NOVALUE =
;  311     0678  1
;  312     0679  1 !++
;  313     0680  1 ! FUNCTIONAL DESCRIPTION:
;  314     0681  1 !
;  315     0682  1 !       This routine takes a g float number and formats it without
;  316     0683  1 !       leading or trailing spaces, and no E format, 15 significant digits,
;  317     0684  1 !       and gives that value to the destination string.
;  318     0685  1 !       Note that this routine violates the calling standard by accepting
;  319     0686  1 !       and passing g floating numbers by value
;  320     0687  1 !
;  321     0688  1 ! FORMAL PARAMETERS:
;  322     0689  1 !
;  323     0690  1 !       STRING.wt.dx            pointer to input string descriptor
;  324     0691  1 !       VALUE.rg.v              value of a double number
;  325     0692  1 !       (VALUE1 and VALUE2 used to pick up the 2 longwords of g float value)
;  326     0693  1 !
;  327     0694  1 ! IMPLICIT INPUTS:
;  328     0695  1 !
;  329     0696  1 !       NONE
;  330     0697  1 !
;  331     0698  1 ! IMPLICIT OUTPUTS:
;  332     0699  1 !
;  333     0700  1 !       NONE
;  334     0701  1 !
;  335     0702  1 ! ROUTINE VALUE:
;  336     0703  1 ! COMPLETION CODES:
;  337     0704  1 !
;  338     0705  1 !       NONE
;  339     0706  1 !
;  340     0707  1 ! SIDE EFFECTS:
;  341     0708  1 !
;  342     0709  1 !       This routine calls the conversion routine and so may signal any of its
;  343     0710  1 !       errors or have any of its side effects.  In particular, the conversion
;  344     0711  1 !       routine calls STR$ routines and so may allocate or deallocate
;  345     0712  1 !       dynamic string space and write lock a string for a short time.
;  346     0713  1 !
;  347     0714  1 !--
;  348     0715  1
;  349     0716  2     BEGIN
;  350     0717  2
;  351     0718  2     MAP
;  352     0719  2         STRING : REF BLOCK [8,BYTE];
;  353     0720  2
;  354     0721  2     LOCAL
;  355     0722  2         STR_LENGTH : WORD;                      ! conversion rtn returns len
;  356     0723  2
;  357     0724  2     BAS$CVT_OUT_G_F (VALUE1,                    ! convert this value to string
;  358     0725  2                      0,                         ! don't specify integer digits
;  359     0726  2                      0,                         ! don't specify fraction digits
;  360     0727  2                      strip_spaces + floating_dec, ! set flags
;  361     0728  2                      STR_LENGTH,                ! return bytes needed for str
;  362     0729  2                      STRING [0,0,0,0]);         ! return string
```

```
;  363          0730  2
;  364          0731  2     RETURN;
;  365          0732  1     END;                                    !End of BAS$NUM1_G



                                    0000 00000        .ENTRY  BAS$NUM1_G, Save nothing                ; 0673
                              5E          04   C2 00002  SUBL2   #4, SP                               ;
                           04     AC   DD 00005  PUSHL   STRING                                      ; 0729
                           04     AE   9F 00008  PUSHAB  STR_LENGTH                                  ; 0724
                                  21   DD 0000B  PUSHL   #33                                        ; 0729
                                  7E   7C 0000D  CLRQ    -(SP)                                       ;
                           08     AC   9F 0000F  PUSHAB  VALUE1                                      ; 0724
                  00000000G  00          06   FB 00012  CALLS   #6, BAS$CVT_OUT_G_F                  ; 0729
                                         04 00019  RET                                               ; 0732

; Routine Size:  26 bytes,    Routine Base:  _BAS$CODE + 0062
```

```
;  367     0733  1 GLOBAL ROUTINE BAS$NUM1_H (                  ! convert h float to string
;  368     0734  1                             STRING,           ! Address of destination descriptor
;  369     0735  1                             VALUE1,           ! 1st longword of h float value to put in
;  370     0736  1                             VALUE2,           ! 2nd longword of h float value for string
;  371     0737  1                             VALUE3,           ! 3rd longword of h float value
;  372     0738  1                             VALUE4) :         ! 4th longword of h float value
;  373     0739  1                             NOVALUE =
;  374     0740  1
;  375     0741  1 !++
;  376     0742  1 ! FUNCTIONAL DESCRIPTION:
;  377     0743  1 !
;  378     0744  1 !       This routine takes an h float number and formats it without
;  379     0745  1 !       leading or trailing spaces, 33 significant digits,
;  380     0746  1 !       and gives that value to the destination string.
;  381     0747  1 !       Note that this routine violates the calling standard by accepting
;  382     0748  1 !       and passing h floating numbers by value
;  383     0749  1 !
;  384     0750  1 ! FORMAL PARAMETERS:
;  385     0751  1 !
;  386     0752  1 !       STRING.wt.dx            pointer to input string descriptor
;  387     0753  1 !       VALUE.rh.v              value of an h floating number
;  388     0754  1 !       (VALUE1, VALUE2, VALUE3, and VALUE4 used to pick up the 4 longwords of h float value)
;  389     0755  1 !
;  390     0756  1 ! IMPLICIT INPUTS:
;  391     0757  1 !
;  392     0758  1 !       NONE
;  393     0759  1 !
;  394     0760  1 ! IMPLICIT OUTPUTS:
;  395     0761  1 !
;  396     0762  1 !       NONE
;  397     0763  1 !
;  398     0764  1 ! ROUTINE VALUE:
;  399     0765  1 ! COMPLETION CODES:
;  400     0766  1 !
;  401     0767  1 !       NONE
;  402     0768  1 !
;  403     0769  1 ! SIDE EFFECTS:
;  404     0770  1 !
;  405     0771  1 !       This routine calls the conversion routine and so may signal any of its
;  406     0772  1 !       errors or have any of its side effects.  In particular, the conversion
;  407     0773  1 !       routine calls STR$ routines and so may allocate or deallocate
;  408     0774  1 !       dynamic string space and write lock a string for a short time.
;  409     0775  1 !
;  410     0776  1 !--
;  411     0777  1
;  412     0778  2     BEGIN
;  413     0779  2
;  414     0780  2     MAP
;  415     0781  2         STRING : REF BLOCK [8,BYTE];
;  416     0782  2
;  417     0783  2     LOCAL
;  418     0784  2         STR_LENGTH : WORD;                      ! conversion rtn returns len
;  419     0785  2
;  420     0786  2     BAS$CVT_OUT_H_F (VALUE1,                    ! convert this value to string
;  421     0787  2                      0,                         ! don't specify integer digits
;  422     0788  2                      0,                         ! don't specify fraction digits
;  423     0789  2                      strip_spaces + floating_dec, ! set flags
```

```
;  424          0790  2                    STR_LENGTH,                 ! return bytes needed for str
;  425          0791  2                    STRING [0,0,0,0]);          ! return string
;  426          0792  2
;  427          0793  2        RETURN;
;  428          0794  1        END;                                      !End of BAS$NUM1_H



                                        0000 00000      .ENTRY  BAS$NUM1_H, Save nothing                ; 0733
                                5E        04  C2 00002   SUBL2   #4, SP                                ;
                             04   AC      DD 00005      PUSHL   STRING                                  ; 0791
                             04   AE      9F 00008      PUSHAB  STR_LENGTH                              ; 0786
                                  21      DD 0000B      PUSHL   #33                                     ; 0791
                                  7E      7C 0000D      CLRQ    -(SP)                                   ;
                             08   AC      9F 0000F      PUSHAB  VALUE1                                  ; 0786
                00000000G    00        06  FB 00012   CALLS   #6, BAS$CVT_OUT_H_F                      ; 0791
                                         04 00019      RET                                             ; 0794

; Routine Size:  26 bytes,    Routine Base:  _BAS$CODE + 007C
```

```
;  430          0795  1 GLOBAL ROUTINE BAS$NUM1_P (                  ! convert packed to string
;  431          0796  1                             STRING,           ! Address of destination descriptor
;  432          0797  1                             VALUE) :          ! Create string with this value
;  433          0798  1                             NOVALUE =
;  434          0799  1
;  435          0800  1 !++
;  436          0801  1 ! FUNCTIONAL DESCRIPTION:
;  437          0802  1 !
;  438          0803  1 !       This routine takes a packed decimal number and formats it
;  439          0804  1 !       with no leading or trailing spaces and no E format, 6 significant
;  440          0805  1 !       digits, and gives that value to the destination string.
;  441          0806  1 !
;  442          0807  1 ! FORMAL PARAMETERS:
;  443          0808  1 !
;  444          0809  1 !       STRING.wt.dx            pointer to input string descriptor
;  445          0810  1 !       VALUE.rp.dsd            desc of a packed decimal number
;  446          0811  1 !
;  447          0812  1 ! IMPLICIT INPUTS:
;  448          0813  1 !
;  449          0814  1 !       NONE
;  450          0815  1 !
;  451          0816  1 ! IMPLICIT OUTPUTS:
;  452          0817  1 !
;  453          0818  1 !       NONE
;  454          0819  1 !
;  455          0820  1 ! ROUTINE VALUE:
;  456          0821  1 ! COMPLETION CODES:
;  457          0822  1 !
;  458          0823  1 !       NONE
;  459          0824  1 !
;  460          0825  1 ! SIDE EFFECTS:
;  461          0826  1 !
;  462          0827  1 !       This routine calls the conversion routine and so may signal any of its
;  463          0828  1 !       errors or have any of its side effects.  In particular, the conversion
;  464          0829  1 !       routine calls STR$ routines and so may allocate or deallocate
;  465          0830  1 !       dynamic string space and write lock a string for a short 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;                      ! cvt returns str lengtht
;  477          0842  2
;  478          0843  2     BAS$CVT_OUT_P_F (.VALUE,                    ! convert this value to string
;  479          0844  2                      0,                         ! don't specify integer digits
;  480          0845  2                      0,                         ! don't specify fraction digits
;  481          0846  2                      strip_spaces + floating_dec, ! set flags
;  482          0847  2                      STR_LENGTH,                ! return bytes needed for str
;  483          0848  2                      STRING [0,0,0,0]);         ! return string
;  484          0849  2
;  485          0850  2     RETURN;
;  486          0851  1     END;                                        !End of BAS$NUM1_P
```

```
                                 0000 00000     .ENTRY  BAS$NUM1_P, Save nothing                ; 0795
                          5E       04 C2 00002  SUBL2   #4, SP                                 ;
                       04 AC       DD 00005     PUSHL   STRING                                 ; 0848
                       04 AE       9F 00008     PUSHAB  STR_LENGTH                             ; 0843
                          21       DD 0000B     PUSHL   #33                                    ; 0848
                          7E       7C 0000D     CLRQ    -(SP)                                  ;
                       08 AC       DD 0000F     PUSHL   VALUE                                  ;
                00000000G  00      06 FB 00012  CALLS   #6, BAS$CVT_OUT_P_F                    ;
                                    04 00019     RET                                            ; 0851

; Routine Size:  26 bytes,    Routine Base:  _BAS$CODE + 0096
```

```
;  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$:BASNUM1/OBJ=OBJ$:BASNUM1 MSRC$:BASNUM1/UPDATE=(ENH$:BASNUM1)

; Size:         176 code + 0 data bytes
; Run Time:        00:06.7
; Elapsed Time:    00:15.0
; Lines/CPU Min:    7673
; Lexemes/CPU-Min: 19889
; Memory Used:  39 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